

2019

## Adaptive Strategies of Multi-Objective Optimization for Greener Networks

Hatem Yazbek

Follow this and additional works at: [https://nsuworks.nova.edu/gscis\\_etd](https://nsuworks.nova.edu/gscis_etd)



Part of the [Computer Sciences Commons](#)

## Share Feedback About This Item

---

This Dissertation is brought to you by the College of Computing and Engineering at NSUWorks. It has been accepted for inclusion in CCE Theses and Dissertations by an authorized administrator of NSUWorks. For more information, please contact [nsuworks@nova.edu](mailto:nsuworks@nova.edu).

ADAPTIVE STRATEGIES OF MULTI-OBJECTIVE OPTIMIZATION FOR  
GREENER NETWORKS

By

Hatem Yazbek

A dissertation report submitted in partial fulfillment of the requirements  
for the degree of Doctor of Philosophy  
in  
Computer Science

College of Computing and Engineering  
Nova Southeastern University

2019

We hereby certify that this dissertation, submitted by Hatem Yazbek conforms to acceptable standards and is fully adequate in scope and quality to fulfill the dissertation requirements for the degree of Doctor of Philosophy.



Peixiang Liu, Ph.D.  
Chairperson of Dissertation Committee

12/4/2019

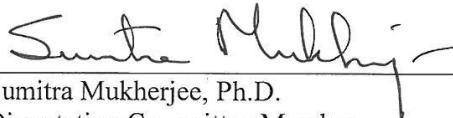
Date



Michael J. Laszlo, Ph.D.  
Dissertation Committee Member

12/4/19

Date



Sumitra Mukherjee, Ph.D.  
Dissertation Committee Member

12/4/2019

Date

Approved:



Meline Kevorkian, Ed.D.  
Interim Dean, College of Computing and Engineering

12/9/2019

Date

College of Computing and Engineering  
Nova Southeastern University

2019

An Abstract of a Dissertation Report Submitted to Nova Southeastern University in  
Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

Adaptive Strategies of Multi-Objective Optimization for Greener Networks

by  
Hatem Yazbek

Increasing energy costs and environmental issues related to the Internet and wired networks continue to be a major concern. Energy-efficient or power-aware networks continue to gain interest in the research community. Existing energy reduction approaches do not fully address all aspects of the problem. We consider the problem of reducing energy by turning off network links, while achieving acceptable load balance, by adjusting link weights. Changing link weights frequently can cause network oscillation or instability in measuring the resulting traffic load, which is a situation to be avoided. In this research, we optimize two objectives, which are minimizing network power consumption by maximizing utilization of shortest paths, and at the same time achieving load-balance by minimizing network Maximum Link Utilization (MLU).

Research to date has focused on the link level of traffic load balance, to minimize power consumption, while putting less focus on utilizing adaptive strategic techniques that optimize multi-objectives problems. This research developed a new approach that relies on live data collected from wired networks, and performs Multi-Objective Optimization (MOO) using a Non-dominated Sorting Genetic Algorithm (NSGA-II) that applies alternative adaptive strategies in order to optimize those two objectives. We also studied how adding delays between link weights adjustments can alleviate the network oscillation or instability without causing higher network power consumption and imbalanced network traffic.

This work introduced a novel approach to select underutilized links to go to sleep using adaptive strategies of MOO that are aware of traffic changes. Re-computing the algorithm takes less than a minute, while network traffic is frequently updated every few minutes. The hybrid approach that we designed was able to reduce the power consumption by 35.24%, while reducing MLU by 42.86% for specific traffic pattern used in Abilene network topology. For network instability, we introduced *sequential\_delay* and *wait\_interval* delay parameters that are implemented in conjunction with link weight settings. We show reduction of instability measurement from 175% down to 8.6% for Abilene network topology when using a value of 1sec for both *sequential\_delay* and *wait\_interval* delay parameters.

**Keywords:** Shortest path, network traffic, link-weight setting, multi-objective optimization, traffic engineering, energy-aware, instability.

## **Acknowledgements**

The strenuous and long journey until I completed my PhD degree was not possible without the support of many people. My utmost gratitude goes to my wife and family, for their support and understanding. I am also grateful for my parents' encouragement, especially my mother (may she rest in peace) who always wanted to see me become a Doctor.

I was inspired to pursue my PhD dissertation by few colleagues who pursued their PhD while keeping their full-time job. Adopting the work of Dr. Frederic Francoise allowed me to develop a very good relationship with him. His constant support and advice in my attempt at re-creating his work accelerated my march into this research field.

Finally, I would like to thank the faculty and staff of NSU's college of computing and engineering and mainly my dissertation chair Dr. Peixiang Liu, for his enormous guidance and support. His reviews of my work gave me valuable advice, especially in enlightening me on methods to overcome obstacles in the thesis. Dr. Sumitra Mukherjee and Dr. Michael Laszlo, thank you for being on my dissertation committee, for providing valuable comments and reviews, and for giving me the opportunity to vastly improve my knowledge through your courses.

## Contents

List of Tables.....	viii
List of Figures .....	ix
Chapter 1 - Introduction .....	10
Background .....	10
Problem Statement .....	11
<i>Energy-aware Networks</i> .....	11
<i>Multi-Objective Optimization of Network Power Consumption and MLU</i> .....	12
<i>Problem Formulation</i> .....	13
<i>Network Traffic and Metrics</i> .....	16
Dissertation Goal .....	17
Research Questions .....	18
Relevance and Significance.....	18
Barriers and Issues.....	19
Limitations and Delimitations of Study .....	20
Definition of Terms .....	21
Summary .....	22
Chapter 2 - Review of Literature.....	24
Management and Schemes of Energy Aware Networks .....	25
<i>Network Energy Efficiency</i> .....	26
<i>Traffic Engineering</i> .....	28
Network Traffic Measurement and Online OSPF .....	29
Routing Instability and Network Oscillation.....	31
Network Energy Aware Approaches and Methods .....	32
<i>Heuristic Approaches</i> .....	32
<i>Genetic Algorithms, and Other</i> .....	34
Summary of Literature .....	37
Chapter 3 - Methodology.....	38
Why NSGA-II for Multi-Objective Optimization? .....	38
<i>Non-dominance technique</i> .....	39

<i>Crowding Distance</i> .....	39
GA Solution Encoding, and Scheme Overview .....	40
An Illustrative Example of Network Energy and MLU MOO Algorithm .....	42
Traffic Engineering Schemes .....	44
Dual Approach 1 (MOGA-I) .....	45
Delta Weight Approach 2 (MOGA-II) .....	46
Hybrid Approach 3 (Hybrid MOGA-III) .....	49
Common Issues in All Three Approaches .....	50
Adaptive GA Genetic Operators .....	50
<i>GA Crossover Operator</i> .....	51
<i>GA Mutation Operator</i> .....	52
Network Instability and Delay Parameters .....	53
<i>Sequential Delay Method</i> .....	54
<i>Delay Wait Interval Method</i> .....	55
Resource Requirements .....	58
Summary of Methodology .....	59
Chapter 4 - Results .....	61
Introduction .....	61
Network Traffic and Data Sets Description .....	61
<i>Network Data Sets and Usage</i> .....	63
<i>Multi-objective DEAP Optimization Parameters Setup</i> .....	65
Simulation Results .....	68
<i><math>\Delta w</math> Simulations for the Delta Weight Approach</i> .....	68
<i>Abilene Network Topology Results</i> .....	70
<i>Polska Network Topology Results</i> .....	73
Network Flow Instability Measurements and Experiments .....	76
<i>Sequential Delay Effect on Instability</i> .....	77
<i>Wait Interval Delay Effect on Instability</i> .....	80
<i><math>\Delta w</math> vs. Instability Measurement</i> .....	84
<i>Final Results Including <math>\Delta w</math> and Best Instability Measurement</i> .....	89
Summary of Results .....	93
Chapter 5 - Conclusions .....	95

Overview .....	95
Conclusions .....	96
<i>Limitation of Network Topology Flow</i> .....	97
Implications .....	97
Recommendations for Future Work .....	98
<i>Network Topology Flow Measurement and Online Traffic</i> .....	98
<i>Larger Network Topologies with Real Time Traffic</i> .....	99
<i>Adaptive GA Strategies of Multi Processing with Learning Methods</i> .....	99
<i>Further Instability Studies with Comprehensive and Variable Delay Parameters</i> .....	99
Summary .....	100



## List of Tables

Table 1 - <i>Summary of Model Notations</i> .....	13
Table 2 - <i>Criteria for link/node sleeping</i> . ....	44
Table 3 - <i>MOO Algorithm with DEAP program best parameters settings</i> .....	66
Table 4 – <i>PC and MLU savings for three sets of traffic, Abilene topology</i> .....	73
Table 5 - <i>PC and MLU Savings for three sets of traffic, Polska Topology</i> .....	76
Table 6 - <i>Final Abilene PC and MLU savings with Hybrid_stable approach</i> .....	92
Table 7 - <i>Final Polska PC and MLU savings with Hybrid_stable approach</i> .....	93

## List of Figures

<i>Figure 1</i> - Energy Usage vs. Link Utilization. ....	27
<i>Figure 2</i> – Network Links Mapping to Chromosome Encoding. ....	41
<i>Figure 3</i> - Illustrative example for network topology link weights optimization. ....	42
<i>Figure 4</i> - Dual Chromosome Approach. ....	45
<i>Figure 5</i> - MOGA-II Two Step Algorithm. ....	47
<i>Figure 6</i> - LSS Aware Crossover Operator. ....	52
<i>Figure 7</i> - Diagram of CORE and Algorithm for Setting Instability Delay Parameters. ....	55
<i>Figure 8</i> - Pseudo Code Showing <i>wait_interval</i> and <i>sequential_delay</i> Algorithm. ....	57
<i>Figure 9</i> – (a) Abilene Topology, (b) Polska Topology. ....	62
<i>Figure 10</i> – (a) Abilene Traffic Data Set from 2004-08-05, (b) Abilene Traffic per Node. ....	64
<i>Figure 11</i> - Example of Pareto Front Solutions with Optimal Solutions. ....	67
<i>Figure 12</i> - MLU and PC vs. Link $\Delta w$ , Abilene Topology. ....	69
<i>Figure 13</i> - MLU and PC vs. Link $\Delta w$ , Polska Topology. ....	70
<i>Figure 14</i> - PC vs. Approach, Abilene Topology. ....	71
<i>Figure 15</i> - MLU vs. Approach, Abilene Topology. ....	72
<i>Figure 16</i> - PC vs. Approach, Polska Topology. ....	74
<i>Figure 17</i> - MLU vs. Approach, Polska Topology. ....	75
<i>Figure 18</i> - Avg. Instability Measure, Abilene Topology, <i>wait_interval</i> =1sec. ....	77
<i>Figure 19</i> - Avg. Instability Measure, Abilene Topology, <i>wait_interval</i> =5sec. ....	78
<i>Figure 20</i> - Avg. Instability Measure, Polska Topology, <i>wait_interval</i> =1sec. ....	79
<i>Figure 21</i> - Avg. Instability Measure, Polska Topology, <i>wait_interval</i> =5sec. ....	80
<i>Figure 22</i> - Avg. Instability Measure, Abilene Topology, <i>sequential_delay</i> =0.1sec. ....	81
<i>Figure 23</i> - Avg. Instability Measure, Abilene Topology, <i>sequential_delay</i> =1sec. ....	82
<i>Figure 24</i> - Avg. Instability Measure, Polska Topology, <i>sequential_delay</i> =0.1sec. ....	83
<i>Figure 25</i> - Avg. Instability Measure, Polska Topology, <i>sequential_delay</i> =1sec. ....	84
<i>Figure 26</i> - MLU and PC vs. $\Delta w$ , with Delay and Instability, Abilene Topology. ....	85
<i>Figure 27</i> - MLU and PC vs. $\Delta w$ with Delay and Instability, Polska Topology. ....	86
<i>Figure 28</i> – Instability, MLU and PC vs. $\Delta w$ with Zero Delay, Abilene Topology. ....	87
<i>Figure 29</i> - Instability, MLU and PC vs. $\Delta w$ with Zero Delay, Polska Topology. ....	88
<i>Figure 30</i> - PC Comparison of Hybrid Approaches with Instability, Abilene Topology. ....	89
<i>Figure 31</i> - MLU Comparison of Hybrid Approaches with Instability, Abilene Topology. ....	90
<i>Figure 32</i> - PC Comparison of Hybrid Approaches with Instability, Polska Topology. ....	91
<i>Figure 33</i> - MLU Comparison of Hybrid Approaches with Instability, Polska Topology. ....	92

## Chapter 1 - Introduction

### Background

A considerable amount of recent research is focused on how to save energy in Internet Protocol (IP) networks while satisfying the network traffic demands and controlling its load balance. The internet network power consumption is rapidly growing and is responsible for more than 5% of the total energy used in the world (Addis, Capone, Carello, Gianoli, & Sanso, 2012). Many researchers approach this problem by proposing traffic management methods of saving energy using heuristic approach, and based on specific time periods of the network traffic. A heuristic approach is used for diverting the traffic from underutilized links into other links; therefore make those underutilized links go to sleep. This research work focuses on continuous dynamic sampling of the network traffic state across time and applies an adaptive Multi-Objective Optimization (MOO) method in order to achieve reduced power consumption and acceptable link utilization in IP network.

Research to date with respect to energy aware network optimization relies on measuring network traffic and matrices that are not dynamic nor changing as described in (Addis et al., 2012; Chiaraviglio, Mellia, & Neri, 2012). Other research implemented an online approach in order to cover dynamic changing network traffic but they do not consider energy aware concern as they focus on reducing routing congestion (Vallet & Brun, 2014). Observing the prevailing traffic changes and adapting the optimization algorithm accordingly is what this research had attempted by using MOO with adaptive strategies.

This dissertation report is organized as follows: First, the problem statement and formalization are presented in Chapter 1. Then the goal and questions of this research

along with its significance and relevance is discussed. In addition, barriers and issues that this research needs to overcome are described. In Chapter 2, a review of literature is compared and synthesized. In Chapter 3, we describe the methodology of this research, including the methods and techniques of applying multi-objective optimization techniques in order to minimize both power consumption and Maximum Link Utilization (MLU). We also examine and present a study and methodology of measuring and reducing network instability when altering link weights. A brief description of resource requirements is given in the end of Chapter 3. Chapter 4 contains the results of numerous experiments performed on two network topologies, including traffic flow inputs, and sensitivity analysis of the wait and delay times. Chapter 5 concludes this dissertation, includes a discussion on how we achieved the research goals, and presents some future research opportunities.

## **Problem Statement**

### *Energy-aware Networks*

Increasing energy costs and environmental issues related to internet and wired networks are becoming a major concern (Bianzino, Chaudet, Rossi, & Rougier, 2012). Reducing power consumption of network infrastructure and elements in the internet has been the subject of recent research. Those studies focus on saving energy based on redundant network links and routers that are less used or under light traffic loads. The network configurations can be re-computed in order to save energy (Vasić et al., 2011). The prime network infrastructure, such as routers, switches, and other devices, still lacks effective energy management solutions (Zhang, Yi, Liu, & Zhang, 2010). Reducing power consumption has been an important part of networking research with less focus on

wired networks (Fisher, Suchara, & Rexford, 2010). By 2020, European Telecoms are expected to consume around 35.8TWh of total network power annually, where the contribution of backbone networks out of total network power consumption will increase from 10% to 40% by 2020 (Francois, Wang, Moessner, Georgoulas, & Xu, 2014).

In order to tackle the increase in network power consumption, different approaches for managing the network load balance have been proposed such as GreenTE traffic engineering mechanism, Interior Gateway Protocol Weight Optimization (IGP-WO), and Green Load-balancing Algorithm (GLA) by (Amaldi et al., 2013; Francois et al., 2014; Zhang et al., 2010). A limited number of recent research works as shown in Energy-Aware Routing (EAR) research by Bianzino, Chaudet, Larroca, Rossi, and Rougier (2010) have been dedicated to energy-aware traffic engineering.

#### *Multi-Objective Optimization of Network Power Consumption and MLU*

In order to find optimal solutions to energy-aware routing, and to find candidate links that can be turned off or put to sleep, most recent approaches considered by researchers rely on heuristic methods as in (Fisher et al., 2010; Heller et al., 2010). The majority of researchers formulate the energy-aware problem as Mixed Integer Linear Programming (MILP) models (Zhang et al., 2010). Computing optimal solutions to energy-aware routing tables is an NP-hard problem (Vasić et al., 2011). In this research we found good solutions to a multi-objective function that models both power consumption and MLU using evolutionary optimization techniques (Deb, Agrawal, Pratap, & Meyarivan, 2000). The first objective is to reduce power consumption of the network, while the second objective is to reduce the MLU as part of achieving network load-balance. The objective of energy reduction is achieved by using a set of shortest

paths that reduces the number of active links by adjusting their link weights, so network link utilization is increased as fewer links carry the traffic. We developed an algorithm that uses MOO and reduces network power consumption while maintaining network MLU below certain allowed threshold. In MOO, the objective functions are conflicting, and there are a (possibly infinite) number of Pareto-optimal solutions. Each solution that is chosen out of a set of Pareto-optimal solutions presents a different trade-off between MLU and power reduction. The upper limit and lower limit of MLU are selected based on the network traffic and topology, and are pre-determined by network operator. We want the links to be well utilized, but not to exceed an upper limit for MLU. On the other hand, the MLU lower limit is applied in order to reduce the number of solutions being generated. No solutions with an MLU lower than MLU lower limit are considered.

#### *Problem Formulation*

The symbols used for formulating the two objectives are defined in Table 1.

Table 1 - *Summary of Model Notations*

Notation	Description
<b><i>Objective Functions</i></b>	
$MLU$	Maximum Link Utilization
$PC$	Power Consumption
<b><i>Model Parameters</i></b>	
$G(N, L)$	Directed graph with set of nodes $N$ and set of links $L$
$P_m$	Node power consumption of router $m$
$P_{ij}$	Power consumption of link $l$ from node $i$ to node $j$
$u_{ij}$	Utilization of link $l$ from node $i$ to node $j$
$c_{ij}$	Bandwidth capacity of link $l$ from node $i$ to node $j$
$D^{sd}$	Traffic demand from node $s$ to node $d$
$\alpha$	Maximum allowable utilization ratio of link capacity
$K$	Set of all $ N ^2$ Source-Destination (SD) pairs
<b><i>Decision Variables</i></b>	
$x_{ij}$	1 if link $l$ from node $i$ to node $j$ is active, 0 otherwise
$y_m$	1 if node $m$ is active, 0 otherwise
$f_{ij}^{sd}$	Traffic flow from node $s$ to node $d$ that traverses link $l$ from node $i$ to node $j$

We model the network as a directed graph  $G = (N, L)$ , where  $N$  is the set of nodes (routers) and  $L$  is the set of links (line cards). We use a directed graph since the amount of network traffic flow can vary based on its direction, meaning a flow from node  $i$  to node  $j$  can be different from the flow from node  $j$  to node  $i$ . A link  $l \in L$  from node  $i$  to node  $j$  can be put to sleep if there is no traffic on the link, and a node  $m \in N$  or a router can be put to sleep if all of its links are asleep (Zhang et al., 2010). Given network topology  $G$  and demand volumes  $D$  for all SD pairs, the model determines network topologies and link weights metrics comprised by active links. Let  $K$  denote the total set of SD pairs, which has  $|N|^2$  number of pairs. For every SD pair  $(s, d) \in K$ , there exists a traffic demand  $D^{sd}$ , and for every link  $l$  from node  $i$  to node  $j$  there is a traffic flow  $f_{ij}^{sd}$ , which is used as a decision variable as shown below.

***Decision variables:***

$x_{ij}$ : Binary decision variable that represents the power status (on/off) of link  $l$  from node  $i$  to node  $j$ . A value of 1 indicates that link  $l$  is active and 0 otherwise.

$y_m$ : Binary decision variable that represents the power status (on/off) of node  $m$  (router). A value of 1 indicates node  $m$  is active and 0 otherwise.

$f_{ij}^{sd}$ : Traffic flow from node  $s$  to node  $d$  traversing link  $l$  from node  $i$  to node  $j$ .

***Objective functions:***

Joint optimization of network power using shortest paths and shutting down as many links and nodes as possible, while minimizing link utilization in order to achieve load balancing, can be formulated as a MILP with the following two objectives:

$$\text{minimize } PC = \sum_{(i,j) \in L} P_{ij} x_{ij} + \sum_{m=1}^{|N|} P_m y_m \quad (1)$$

$$\text{minimize } MLU = \max \left( \frac{1}{c_{ij}} \sum_{(s,d) \in K} f_{ij}^{sd} \right) \quad \forall (i,j) \in L \quad (2)$$

*Subject to following constraints:*

$$\sum_{j=1}^{|N|} f_{ij}^{sd} - \sum_{j=1}^{|N|} f_{ji}^{sd} = \begin{cases} D^{sd} & \forall s, d, i = s \\ -D^{sd} & \forall s, d, i = d \\ 0 & \forall s, d, i \neq s, d \end{cases} \quad (3)$$

$$\sum_{(s,d) \in K} f_{ij}^{sd} \leq \alpha x_{ij} c_{ij} \quad \forall (i,j) \in L, \alpha \in [0,1] \quad (4)$$

$$\sum_{(s,d) \in K} f_{ij}^{sd} \geq 0 \quad \forall (i,j) \in L \quad (5a)$$

$$x_{ij} = 1 \text{ iff } \sum_{(s,d) \in K} f_{ij}^{sd} > 0, \forall (i,j) \in L \quad (5b)$$

$$y_m = 0 \text{ iff } \left( \sum_{(s,d) \in K} \sum_{(i,m) \in L} f_{im}^{sd} + \sum_{(s,d) \in K} \sum_{(m,j) \in L} f_{mj}^{sd} \right) = 0$$

$$\forall m \in [1, |N|] \quad (5c)$$

Equation 1 represents the first objective of MOO which minimizes the power consumption (PC) by moving traffic to shortest paths (fewer devices to turn on thus reducing power). We measure the power consumption of the network as the total power consumption of active links and nodes at a specific time snapshot, and periodically sample the network power state after we execute our MOO. Equation 2 represents the second objective of our MOO which minimizes the MLU of the network in order to achieve load-balancing. The utilization of each link is calculated as the sum of all network flow through link  $l$  flowing from node  $i$  to node  $j$ , divided by the link's specified capacity. Equation 3 represents the standard flow conservation constraint. Equation 4 determines that whenever a shortest path topology is used, all active links should have their utilization below a specific threshold  $\alpha$ , and forcing the flow to 0 if the link  $l$  from node  $i$  to node  $j$  is powered off. So we must ensure that when a set of links is put to sleep mode, the MLU does not exceed the link capacity threshold  $\alpha$  when link is active and it is



0 when the link is sleeping. The flow on link  $l$  from node  $i$  to node  $j$  is the sum of all flows for every SD pair that traverses link  $l$ . The flow  $f_{ij}^{sd}$  exists only if the decision variable  $x_{ij}$  is 1. The SD pair from node  $s$  to node  $d$  is a subset of all SD pairs  $\in K$ , which go through link  $l$  from node  $i$  to node  $j$ . Equation 5a states the traffic flow carries a value equal or greater than zero, while equation 5b ensures that the link  $l$  from node  $i$  to node  $j$  is operated if and only if its flow does carry non-zero traffic, and thus  $x_{ij} = 1$ . Equation 5c constrains node  $m$  to be inactive if and only if the sum of all traffic flowing through it is zero, and thus  $y_m = 0$ . Both  $x_{ij}$  and  $y_m$  are binary decision variables that can be set to 0 or 1 values.

### *Network Traffic and Metrics*

Traffic metrics and network measurement methods are crucial for understanding network behavior, and for proposing effective solutions for saving energy (Gupta & Singh, 2003). Researchers must be aware of the response time for waking the network device after putting it to sleep, and must account for traffic changes. One of the difficulties in coming up with a valid solution or approach is that traffic patterns change very frequently (Heller et al., 2010). A concept for measuring traffic, called traffic matrices is introduced in Zhang et al. (2010), where the authors demonstrate that a small number of those matrices are enough to perform network analysis. The authors propose a complementary approach to utilize power management at the network level, by routing traffic through various paths in order to adjust the workload on individual links or routers. They base their power saving assumptions on the fact that high path redundancy and low link utilization exist in today's networks. Their intra-domain traffic engineering mechanism is called GreenTE, which maximizes the number of links that can be put into

sleep under link utilization and packet delay constraints. However, doing so requires frequent adjustment of network routing traffic status and raises the question of how often the adjusting of the traffic matrix is required.

### **Dissertation Goal**

The goal of this research is to develop adaptive strategies for a multi-objective genetic algorithm such that objectives of both reduced network power consumption and load balance are achieved. The Pareto-optimal front arises due to the presence of two objectives in the problem formulation (Francois et al., 2014). The first objective for minimizing power consumption was achieved by increasing link utilization of the shortest paths, and putting underutilized links to sleep. In return, the second objective of load-balance by minimizing MLU was affected, and gave rise to the Pareto-optimal front.

This work achieved the goal of this research by using real time network traffic generated by Multi-Generator (MGEN), and by using similar traffic networks as in (Vasić et al., 2011; Zhang et al., 2010). We performed the following research:

- Investigated network traffic resources available for simulating a real network.
- Explored network traffic engineering techniques, measurement and analysis of network metrics, and link weights management.
- Investigated and developed alternative GA operators and strategies for MOO using NSGA-II, in order to find the Pareto-optimal front of a solution space.
  - Pareto-optimal solutions are selected such that load balance or MLU must meet  $< 50\%-80\%$ , while power consumption is reduced by  $30\%-40\%$  (Francois et al., 2014; Zhang et al., 2010).
- Used and tested adaptive strategies for MOO based on various traffic states

- Traffic matrices are measured every 5 to 15 min. (Zhang et al., 2010)
- We targeted our MOO algorithm to take less than one minute of runtime, while it runs every 5 to 15 minutes.

### **Research Questions**

To address the above problem, several research questions were raised.

1. Can network power consumption be reduced by applying our approach?
2. How much can we tune network load-balance or increase MLU in order to save more power, and can network traffic load be balanced?
3. How to change the network traffic link weights considering observed network online status and how often is adjusting of traffic matrix required?
4. Can this research work for reducing network energy adapt efficiently to most types of network topologies?
5. Traffic changes can cause oscillations, while concurrently changing routing of network paths by altering link weights on the fly. Changing link weights does not consider if the changes can cause disruption. Can we reduce network disruption when changing link weights?

### **Relevance and Significance**

The resulting mechanism developed in this research can provide further benefit to energy aware network research, and increase the knowledge base related to existing benchmarks and use cases. Most of the latest research on this topic use heuristic methods to find those candidate links that will be turned off or go to sleep, in order to save network energy. Such research is discussed in Zhang et al. (2010), Chiaraviglio et al. (2012), Amaldi et al. (2013), and Moulhierac and Phan (2014), with few research papers

using Multi-Objective Genetic Algorithm (MOGA) for solving the same problem (Francois et al., 2014).

This approach considers dynamic traffic matrices and accounts for the traffic changes with respect to time periods. The closest research that covers dynamic changing matrices was done by Vallet and Brun (2014), while not considering energy savings as its objective and rather focusing on reducing network congestion. Other research considered changing traffic matrices by using robust optimization for reducing energy while using few types of traffic patterns and link weight OSPF settings (Moulierac & Phan, 2014).

The goal of our research work is to address the energy reduction in computer networks using concurrent optimization of both energy reduction and minimizing MLU, which considers the changing traffic matrices. Specifically, our approach extends the Interior Gateway Protocol (IGP) link weight optimization of joint energy efficiency and improving load balance (Francois et al., 2014). Our approach can be adopted and leveraged by network operators for managing large networks, especially when scaled to use distributed computing.

The ability to reconfigure the network topology in a dynamic and agile fashion, by adapting to continuous traffic changes would achieve great benefit in reducing power consumption in IP networks. This method can be migrated and adopted by other types of traffic flows. A possible drawback would be that major changes to the algorithm would be needed.

### **Barriers and Issues**

This is a rather new area of research, with relatively scarce literature. Most of the latest research focuses on heuristic approaches that adhere to a specific traffic

engineering scheme. Few papers relate to using multi-objective optimization technique to tackle the problem except for (Francois et al., 2014). Few papers treat both saving energy and simultaneously consider changing traffic flows and patterns (Capone, Cascone, Gianoli, & Sanso, 2013; Chiaraviglio et al., 2012; Vallet & Brun, 2014).

Obtaining network flow data sets is a difficult task, and most of the research uses the same standard data. Some of the data is even based on older use cases which are of academic type such as Abilene and GEANT network topologies. Another issue is that datasets are not available and synthetic data were used in order to show the approach effectiveness. Furthermore, energy values or power metrics for each network node and link will be difficult to obtain whenever we use a new network topology. Most prior researchers used various existing data sets such as Abilene, Rocket-fuel, synthetic topologies, thus a standard format is not used. A major study of test beds and test cases was certainly required in order to establish a solid experimental database for proving our approach.

It is important to note that the success of this research did not result in a comprehensive energy aware network optimization system. Instead, it results in a component of an autonomous optimization system that can dynamically adapt to network traffic changes with the aim of saving internet energy with minimal effect on MLU. This method can respond quickly to specific traffic changes and events by using an adaptive MOGA algorithm.

### **Limitations and Delimitations of Study**

In this dissertation, the effectiveness of our approach is affected by the quality of the network traffic inputs and topology used, as well as by mimicking a real IP network

traffic that can reliably provide energy aware candidate links that can go to sleep. In addition, results are impacted by quality of Pareto front solutions obtained by the MOO.

The type of networks and traffic generated, as well as representation of dynamic traffic that can change every few minutes, is another factor that influenced the obtained results. It is difficult to capture all different kinds of network traffic scenarios by using few network generators, thereby some network patterns were not studied and our approach was not tested thoroughly.

### Definition of Terms

Term	Definition
Genetic Algorithm	A heuristic search method used in artificial intelligence and computing. It is used for finding optimized solutions to search problems based on the theory of natural selection and evolutionary biology.
GreenTE	Green Traffic Engineering.
Maximum Link Utilization (MLU)	This is the Maximum Link Utilization, which is the maximum utilization of all network links.
Multi-Objective Optimization	An optimization involves more than one objective function to be optimized at the same time. It belongs to the area of multiple criteria decision making.
NSGA-II	A Non-dominated Sorting Genetic Algorithm – II which is a very famous multi-objective optimization algorithm that generates Pareto optimal solutions, or non-dominated solutions.
Open Shortest Path First (OSPF)	A routing protocol for Internet Protocol (IP) networks. It uses a link state routing (LSR) algorithm for traffic routing.
Pareto Optimal Front	It shows the number of solutions (possibly infinite) for the objective functions that are supposed to be conflicting.
	A popular protocol for network management. It is used for collecting information from, and configuring, network

Simple Network Management Protocol (SNMP)	devices, such as servers, switches, and routers on an Internet Protocol (IP) network.
Traffic Matrix	A representation of the traffic demand between a set of origin and destination abstract nodes. An abstract node can consist of one or more network elements.
Traffic Flow	A stream of packets between two end-points that can be characterized in a certain way and denoted as source and destination addresses.
Traffic Monitoring	The process of observing traffic characteristics at a given point in a network and collecting the traffic information for analysis and further action.

### Summary

Energy aware networking is gaining more interest from network operators and researchers, as percentage of energy consumed by Internet networks out of global power consumption is rapidly increasing, and its projected to exceed 20% by 2030 (Idzikowski et al., 2016). This research directly addresses the problem of saving energy in IP networks while keeping the network load balance under control. The method of minimizing both the MLU and PC concurrently using MOO was rarely used in literature.

Research to date yielded methods of using heuristic approach that cannot handle online changes of real traffic and relies on sampling the network at fixed time intervals, such as day/night time periods. Those methods are not satisfactory enough to model a real IP network traffic and thus the optimization algorithm does not yield good solutions. In addition to the above, unfortunately the power consumption of current network device architectures and transmission technologies is almost traffic load independent (Addis, Capone, Carello, Gianoli, & Sanso, 2014).

This research introduces a new approach for optimizing both objectives of reducing MLU and reducing the energy or power consumed by the network, by using MOO along with customized GA operators, that are part of adaptive strategies which are aware of the dynamic network traffic.



## Chapter 2 - Review of Literature

The usage of Information and Communication Technology (ICT) and its impact on the environment have fueled a lot of interest among researchers, manufacturers, educators and designers (Zeadally, Khan, & Chilamkurti, 2012). Several recent techniques and solutions have been proposed to minimize power consumption by network devices, protocols, networks, end user systems, and data centers. The energy efficiency issue has also become a high priority objective for wired networks and service infrastructures (Bolla, Bruschi, Davoli, & Cucchietti, 2011). Because of the potential economic benefits and its expected environmental impact, reducing power consumption is becoming a major concern in wired networks (Bianzino et al., 2012). Therefore green networking has been drawing a lot of attention in the last years, emphasized in surveys of (Addis, Capone, Carello, Gianoli, & Sansò, 2016; Bianzino et al., 2012; Bolla et al., 2011).

There are several interesting investigations into different techniques that are used for reducing power and energy with regards to IP networks. This review of literature does not attempt to cover all of the work conducted, but instead this chapter attempts to cover some of the accomplishments made as they relate to the problem of reducing power consumption in wired IP networks. The remainder of this chapter discusses the management and schemes for energy aware networks, network energy efficiency, traffic engineering, online traffic measurements, routing instability and network oscillation, and it gives an overview of some of energy aware approaches and methods that are relevant to this work.

## **Management and Schemes of Energy Aware Networks**

Green networking strategies are described in detail in the work of (Bianzino et al., 2012). The authors individualize four classes of power-aware solutions, namely resource consolidation, virtualization, selective connectedness, and proportional computing. The authors do separate online solutions that act on runtime from offline solutions that act before runtime. They classify the green network research into categories of Adaptive Link Rate (ALR), interface proxying, energy aware infrastructure, and applications. Most research of router power management is focused on component or link levels, treating routers as isolated devices. The authors propose a complementary approach to utilize power management at the network level, by routing traffic through various paths in order to adjust the workload on individual links or routers. They base their power saving assumptions on the fact that high path redundancy and low link utilization exist in today's networks. Their intra-domain traffic engineering mechanism is called GreenTE, which maximizes the number of links that can be put into sleep under link utilization and packet delay constraints. However, doing so requires frequent adjustment of network routing traffic status and raises the question of how often adjusting of the traffic matrix is required.

The idea of energy conservation in internet networks was first published in a position paper by (Gupta & Singh, 2003). It suggests that components of network devices can be put to sleep in a coordinated manner with some changes in internet protocols. Chabarek et al. (2008) extend this idea by using a more coarse-grained network design and routing approach that uses mixed integer optimization techniques to investigate power consumption. Other researchers such as Fisher et al. (2010) combine

benefits of both coordinated and uncoordinated sleeping methods by depending on pre-calculations of the best configuration.

Fisher et al. (2010) address reducing power consumption by exploiting the fact that many links in core networks are actually “bundles” of multiple physical cables and line cards that can be shut down independently. Removing entire links during periods of low demands, from topology reduces capacity but may cause disruptions in the routing protocol. Garroppo, Giordano, Nencioni, and Scutellà (2013b) analyze a novel network-wide power management problem, called PARND-B. They consider both traffic throughput of the nodes and their power consumption in order to turn off both the chassis and Physical Interface Card (PIC). The chassis is the frame that houses all routers and switch circuits inside it. They evaluate their solution on real network scenarios while considering time to obtain solution, power of network elements, traffic load, Quality of Service (QoS) requirement, and number of routing paths for each traffic demand.

#### *Network Energy Efficiency*

Ethernet is the dominant wireline technology for LANs and is widely used in residences and in commercial buildings. A working group started an effort back in 2006 in order to improve what they called Energy Efficient Ethernet (EEE), which became an IEEE standard called 802.3az-2010 (Christensen et al., 2010). Packet coalescing saves energy by aggregating traffic in burst like fashion, instead of sending small network packets. EEE performance can be improved by packet coalescing in which a FIFO (First In First Out) queue is used to collect or coalesce many packets before sending them to a certain link as a back-to-back burst of packets. The main motivation of developing this 802.3az standard is to reduce power consumption of Ethernet interfaces that is widely

used in wired IP networks, and thus obtain large energy savings. The most important fact indicated by authors is that the power consumption of an idle link is 10 percent of that of active link. Figure 1 shows 5 graphs describing the relationship of power or energy use versus link utilization percentage (Christensen et al., 2010). For no EEE case, the constant power usage would be 100 percent and its independent of link utilization.

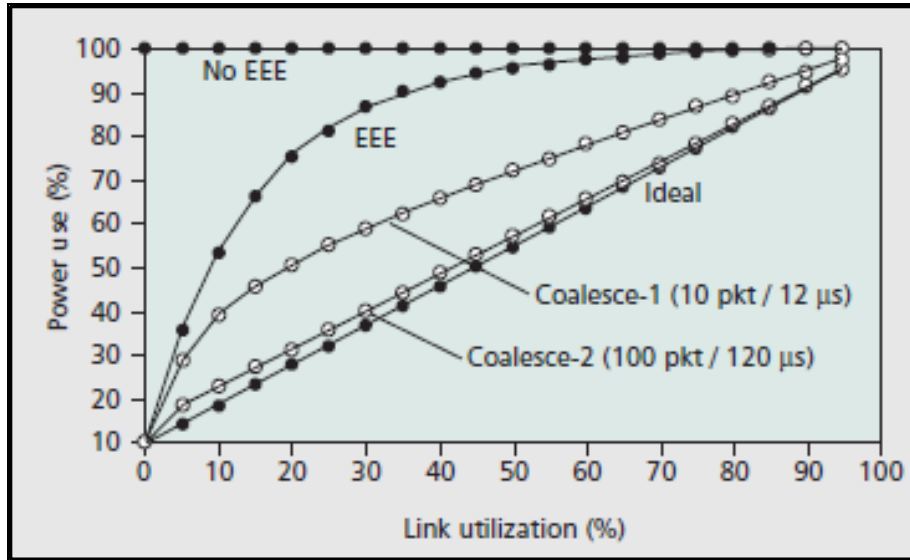


Figure 1 - Energy Usage vs. Link Utilization.

The graph labeled as ideal shows the ideal case where power use is directly proportional to utilization with an offset of 10 percent power consumption in idle case. The first packet arriving to an empty queue initiates a timer (set to  $t_{coalesce}$ ) and a packet counter, and if the maximum packet count ( $max$ ) is reached or the timer expires, then all packets in the coalescing queue are sent. The EEE graph shows power use of EEE without coalescing, while the two remaining graphs labeled as coalesce-1 and coalesce-2 show the results for coalescing with  $t_{coalesce} = 12\mu s$ , and  $max = 10$  packets, and with  $t_{coalesce} = 120\mu s$ , and  $max = 100$  packets, respectively. We can see that coalescing improves energy efficiency while increasing packet delay, and with coalesce-2 the energy

efficiency is closer to ideal. We learn from this analysis, that power usage is proportional to link utilization, while idle links do consume power and it is worthwhile turning them off and make them go to sleep in order to save energy.

### *Traffic Engineering*

Gupta and Singh (2003) identified first the power traffic patterns and how traffic can be routed using energy aware Traffic Engineering (TE). They offered a powering down approach that prompted several researchers to search for new methods. The authors quantify some of the savings that are possible due to inter-packet gaps (i.e., packets not continuously arriving at full speed).

Energy aware networking or green networking follows certain traffic engineering (TE) schemes. Network nodes adhere to different sorting rules, such as random (R), least-link (LL), least-flow (LF), and most-power (MP) as described in (Chiaraviglio et al., 2012). For example, the least-link (LL) heuristic sorts the nodes according to number of links that are sourced or sunk at each node, so nodes with a small number of links are considered first. The least-flow (LF) heuristic considers first those nodes with smallest amount of traffic flowing through them, and most-power (MP) heuristic switches those nodes with the highest power consumption first. In their approach, they change the heuristics based on time period, so for day time they use LF or MP TE scheme. The best energy savings is obtained when using a heuristic algorithm that checks iteratively if a given node or link can be turned off, while sorting nodes first based on a specific TE scheme such as MP. Nodes are turned off first since their energy savings are higher than when switching off single links (Gunaratne, Christensen, & Nordman, 2005).

## Network Traffic Measurement and Online OSPF

Network traffic metrics and measurements remain the most critical issue in evaluating energy efficiency and consumption of devices, hardware, software and various networking architectures, systems and communication protocols (Zeadally et al., 2012). We need to standardize a set of energy-aware metrics with finer granularity. A concept for measuring traffic, called traffic matrices was introduced in (Zhang et al., 2010). They demonstrated that a small number of those matrices are enough to perform network analysis. Considering energy aware optimization when traffic pattern changes in multi-period fashion, where each time period corresponds to a traffic scenario, with inter-period constraints for IP networks was first introduced by (Addis et al., 2014). The time periods (multi-periods) corresponding to different traffic scenarios need to be jointly considered in order to take into account the constraints on routing. Within every time period (inter-period) we can exhibit different routing constraints such as QoS (maximum utilization and maximum path length) and maximum number of link cards switching. The routing can vary according to the different demand conditions, and power consumption is minimized while satisfying the demands in every time interval. They developed a comprehensive Mixed Integer Linear Programming (MILP) formulation of the problem while optimization constraints guarantee that each demand is transported from its origin to its destination.

Mahadevan, Sharma, Banerjee, and Ranganathan (2009) show that actual energy consumed by switches and routers depends on various factors, such as device configurations and traffic workload. Energy and power vary significantly with different configuration settings, and thus these settings are needed when performing power

measurements. They also show that ideal consumption of power is about 70-90% of the maximum power so it is not energy proportional. The authors developed a benchmark that can be used to compare power for a variety of network devices, as they measured power using meters attached to each device. It is a very difficult and inconsistent method to use, where in our approach we rely on power consumption extracted from real network paths and we often re-calculate its power behavior as traffic changes frequently.

Optimizing link weights for energy efficiency in a changing world while considering stability in routing configuration is addressed in (Moulierac & Phan, 2014). Authors focus on the most widely used IGP in IP networks, named OSPF protocol, and utilize a link state approach that performs a calculation of shortest paths based on a set of link weights. They use real-life network data and traffic traces, and aim to reduce serious oscillation in network caused by its reconfiguration. They proposed a robust heuristic optimization with MILP and used the min-load link criteria in order to make the link sleep. Another work that focuses on saving energy under routing domains where OSPF is adopted is shown in (Amaldi et al., 2013). They came up with an offline traffic engineering strategy that considers the daily time intervals characterized by different traffic levels (night, morning, lunch break, afternoon), and they used greedy heuristic algorithms with a two-stage approach. Furthermore, in order to manage the whole network power consumption in a coordinated fashion, switching links and nodes on and off dynamically should track traffic variations (Addis et al., 2014).

Online OSPF weighting optimization in IP networks is key to addressing the high volatility or instability of traffic patterns, where dynamic and adaptive routing schemes are required in order to react to changing traffic (Vallet & Brun, 2014). Their idea is to

deviate traffic from the most loaded links by using delta weight changes to network links. The work of Vallet and Brun (2014) adopts the idea of using delta change of link weights, and use reducing network congestion as an objective. They used SNMP link counts for estimating the traffic demands. Traffic changes every few minutes, while reconfiguring of the network link assignment takes a few seconds, and thus the green networking algorithm should take few seconds to reach a decision. Combining off-line and on-line techniques to optimize OSPF link weights for an energy aware environment was introduced in (Capone et al., 2013). The authors rely on pre-computed scenarios of network traffic, and on discrete time intervals that are not continuous. In Okonor, Wang, Sun, and Georgoulas (2014) link sleeping and wake up optimization techniques based on OSPF were used for saving energy. They consider links that create no traffic disruption as having high priority to sleep and wake-up in order to reduce traffic instability when it is diverted.

### **Routing Instability and Network Oscillation**

Rapid fluctuation and routing instability in IP networks was addressed earlier by (da Silva & Mota, 2017) . Authors emphasize that three primary effects identify network routing instability; packet loss, delay time for converging the network routing and resource overhead such that CPU, memory and other. (da Silva & Mota, 2017) discusses and presents approached for limiting path exploration approach, where accelerating updates and propagation delays of the network paths can improve network instability and reduce fluctuations.

There are several approaches for measuring instability in computer networks, or convergence delay in the internet (Roughan, Willinger, Maennel, Perouli, & Bush, 2011).



Authors emphasize the need to develop internet models that depend on public internet routing databases that can improve the measurement accuracy of the internet model. They show examples of discovery of oscillation and slow convergence in routing protocols, as well as understanding of route flap dampening. An observation was denoted that when a routing congestion is removed, then not all routing entries return to original values (Varadhan, Govindan, & Estrin, 2000). Authors also indicate that it may not be enough to avoid network instability since transit routers can fail and trigger widespread routing instability. They nevertheless emphasize that routing instability is one of the most pathological problems in the internet that can cause loss of service and service degradation for QoS demanding applications.

Shaikh, Varma, Kalampoukas, and Dube (2000) studied the effect of loss of routing protocol messages caused by routing congestion, which lead to route flaps and routing instabilities. They demonstrate how important selective treatment of routing protocol messages from other traffic by using scheduling and other buffer management policies in the routers, in order to achieve stable and robust network operation. Due to policy changes, such as link weights alteration in an OSPF network, then the topology changes and convergence process is triggered (da Silva & Mota, 2017). If a stable state is not reached within a specific period, then packet losses and inconsistencies could occur. During such period, routing instability is identified as fluctuation in network reachability.

## Network Energy Aware Approaches and Methods

### *Heuristic Approaches*

Existing approaches rely on heuristic methods in order to compute those energy-aware critical paths, or calculate those links that can be put to sleep. In Zhang et al.

(2010) and Vasić et al. (2011), the authors model the network as a directed graph  $G=(V,E)$ , where  $V$  is a set of routers and  $E$  is the set of links. They show that making online routing decision by finding optimal routing solution while the real traffic is flowing is an NP-hard problem and can be accomplished in a few hours for a medium-sized network. Accordingly, they recommend using pre-established paths that are constructed off-line in order to avoid long path computation time. Although a power down approach is used in Vasić et al. (2011), turning routers off is not the main concern as they identify a few energy-critical paths off-line, install them into network elements, and use an online element to redirect the traffic such as large part of the network enters low power state. Overall, they developed a new energy saving scheme that is based on identifying and using energy-critical path.

Most of the approaches and algorithms found in the latest literature use heuristic approaches based on energy aware traffic criteria or rules as shown in (Amaldi et al., 2013; Fisher et al., 2010; Zhang et al., 2010). In a latest survey authors classify various approaches covering energy efficiency in core networks with respect to optimal formulations and heuristic solutions (Idzikowski et al., 2016).

Heller et al. (2010) have come up with ElasticTree, which optimizes the power consumption of Data Center Networks by turning off unnecessary links and switches during off-peak hours. ElasticTree also models the problem based on the Multi-Commodity Flow (MCF) model, but is focused on Fat-Tree or similar tree-based topologies. ElasticTree considers link utilization and redundancy when calculating the minimum-power network subset, and is implemented using OpenFlow. This approach leverages the tree-based nature of these networks (e.g., a Fat-Tree) and at runtime

computes a minimal set of network elements to carry the traffic. This is similar to our approach where we re-compute energy critical paths to reduce power, which is calculated online or at runtime as traffic changes. A modified version of OSPF protocol that reduces the number of active links by utilizing shared shortest paths trees is shown in (Cianfrani, Eramo, Listanti, Marazza, & Vittorini, 2010). Their mechanism allows sharing a Shortest Path Tree (SPT) between adjacent routers, so the total number of active links can be reduced and in a distributed way. Concentrating network traffic on a minimal subset of network resources was addressed in (Chiaraviglio et al., 2012). In the latter approach, several backbone nodes get aggregated into one node and after finding an optimal solution for the reduced version of the network they have a method to revert the original nodes back. Given the NP-hard formulation, the optimal solution using a heuristic approach and an Integer Linear Programming (ILP) solver is much more viable since number of nodes is reduced.

#### *Genetic Algorithms, and Other*

Use of Genetic Algorithms in solving network energy aware problem is limited in literature, but was successfully introduced in (Francois et al., 2014). They used GA to find the link weights for the joint optimization of load balancing and energy efficiency, and modified the existing traffic engineering scheme for link sleeping based operations in order to improve end-to-end traffic delay performance. Their work is based on sampling a few different traffic matrices (offline analysis), and found that IP networks have regular traffic patterns. In our work, we build upon their work for customizing the GA operators for mutation and crossover, but in addition, we use adaptive GA strategies that are tuned

for handling dynamic and continuous traffic changes. Those GA operators are part of MOO that uses Non-dominated Sorting Genetic Algorithm (NSGA-II).

Adapting link data rates can reduce the power consumption of links and ports by decreasing their high link capacities. The powering down approach provides more energy saving than rate adaption, which is shown from studies of (Heller et al., 2010; Tucker, Baliga, Ayre, Hinton, & Sorin, 2008). A router is turned off completely in order to accomplish higher energy savings. A combined approach of rate-adaption and power-aware routing, to show savings of significant fraction of network power was used by (Antonakopoulos, Fortune, & Zhang, 2010). Nedeveschi, Popa, Iannaccone, Ratnasamy, and Wetherall (2008) introduced the buffer-and-burst approach that shapes traffic into small bursts in order to create bigger opportunities for network components to sleep. They focus on link level solutions, and suggest the idea of rate-adaptation, which adjusts operating rates of links according to the traffic condition. There are link-level solutions that put line-cards to sleep when the link has no traffic; nonetheless, the power saving from opportunistic sleeping is limited by the inter-arrival time of packets.

Adapting the SDN (Software Defined Networking) approach in order to handle energy-aware routing and resource management for large scale Multiple Protocol Label Switching (MPLS) networks is described in (Celenlioglu, Goger, & Mantar, 2014). They developed a controller using Pre-Established Label Switching Paths (PLSPs), which performs load balancing to minimize congestion in paths, and introduce a path virtualization concept. An excellent description of SDN and its structure is found in (Feamster, Rexford, & Zegura, 2013). They define SDN characteristics such as control and data planes and they show how in turn this facilitates network programmability. A

framework and prototype called Responsive Energy-Proportional Networks (REsPoNse) have been introduced in (Vasić et al., 2011). It identifies the small number of energy-critical paths offline, then installs them into network elements, and use an online element for redirecting the traffic, so a large part of the network elements enter a low power state. One major issue with the latest approach is that offline measurements may not be applicable when the network is active and running. Focus on network wide power saving strategies like Power-Aware Routing is described in (Mahadevan et al., 2009). Power-Aware Network Design (PAND) is further discussed with details in (Garroppo, Giordano, Nencioni, & Scutellà, 2013a). The authors combine both power and network with bundled links to achieve Power-Aware Routing and Network Design with Bundled Links (PARND-BL). They examine powering a single Physical Interface Card (PIC) that is faster than powering on a link, since a link is a topological change that alters the network topology, while a single PIC is a local capability.

Gelenbe and Mahmoodi (2011) use smarter networks using Cognitive Packet Network (CPN) protocol, where AI techniques of Random Neural Network (RNN) are used to save power, gather online QoS measurements and discover new routing paths. They offer an Energy Aware Routing Protocol (EARP) that respects QoS for each incoming flow while attempting to minimize power, so their approach has a combined objective, which is similar to that of Nedevschi et al. (2008) who determine whether to perform rate-adaptation or use sleep mode to save energy. Smarter networks have been proposed to improve QoS and provide QoS-driven routing that performs distributed manner self-improvement by learning from experience of special packets (Sakellari, 2009).

## Summary of Literature

We have presented a comprehensive review of the literature related to energy-aware networking or energy efficiency covering several aspects. Latest research focuses on energy or power management systems and approaches for saving energy especially heuristic methods as shown in latest surveys of (Addis et al., 2016; Bianzino et al., 2012; Bolla et al., 2011; Idzikowski et al., 2016). Most of existing management strategies for reducing energy focus on static analysis of offline and online traffic state, without considering the varying nature of Internet IP network traffic. Network instability induced by changing link weights settings has been addressed by several researchers, especially in an OSPF network that triggers a network convergence process (da Silva & Mota, 2017).

There are different optimizations approaches used in order to save energy which rely mostly on heuristic methods, using MILP modeling of the problem (Idzikowski et al., 2016). These techniques do not take any dynamic traffic demands or multi-period traffic into considerations and merely rely on capturing the network state at a specific time.

## Chapter 3 - Methodology

The study builds upon the work of Francois et al. (2014) by using adaptive optimization methods, and by using a real time varying traffic. We developed methods to model the network inputs, and capture the objective functions by using network simulator and emulator that outputs the power and MLU of the network. We considered energy-aware traffic engineering, along with the traffic matrices and traffic flow. The remainder of this chapter describes the MOGA operators, GA encoding of solutions, illustrative example, traffic engineering schemes, the three approaches that we developed, the adaptive GA crossover and mutation operators, network instability algorithm using delay parameters, and the resources required to carry out the experiments of this study. There are two types of datasets that are used corresponding to network traffic demands and network topologies.

The new approach described in this research built upon the work conducted by Francois et al. (2014) by customizing the GA operators for mutation and crossover, but in our approach we further developed adaptive GA operators and strategies that can handle dynamic and continuous traffic changes. This new approach uses MOO, while considering both objectives of reducing power consumption and keeping MLU under a certain limit  $\alpha$  achieving a steady conventional traffic engineering or network load balance performance

### Why NSGA-II for Multi-Objective Optimization?

The MOGA uses NSGA-II algorithm with customized mutation and crossover operators. NSGA-II is a multi-objective genetic algorithm that was first introduced by

(Deb, Pratap, Agarwal, & Meyarivan, 2002). The performance of the algorithm is based on ranking and selecting the population fronts performed by two additional specialized multi-objective operators called non-dominance technique and a crowding distance. The next generation of solutions is created by combination of current population and the offspring generated according to non-dominance and crowding distance techniques. The non-dominated technique and crowding distance are described in the following sub sections.

#### *Non-dominance technique*

For a problem with  $k$  objective functions, the solution  $x_1$  dominates solution  $x_2$  for all the objective functions if solution  $x_1$  is not worse than  $x_2$  and for at least one of the  $k$  objective functions,  $x_1$  is exactly better than  $x_2$ . Pareto front number 1 is made by all solutions that are not dominated by any other solutions and front number 2 is built by all solutions that are only dominated by solutions in front number 1 and the same goes for other fronts. Therefore, we rank the population based on dominance depth, and which front is the individual located.

#### *Crowding Distance*

The crowding distance is used to measure the density of solutions. The crowding distance value is calculated for each individual in the population, and for a specific solution it is the average distance from the two neighboring solutions immediately before and after along each objective dimension. Ranking among all of those belonging to a front is done where higher value of crowding distance is preferred. To select solutions for the crossover and mutation operators, a binary tournament selection procedure is used. First, the procedure selects two solutions from the population, and then selects the



better one according to non-dominated sorting technique and crowding distance value. If both solutions are selected from the same front, the solution with the higher crowding distance is selected.

Optimal solutions to a multi-objective problem are non-dominated solutions, known as Pareto-optimal solutions, where depending on the optimization case, one point is preferred over the other (Srinivas & Deb, 1994). NSGA-II works in a similar way as traditional genetic algorithms, but we chose it since it preserves elitism and diversity of the solutions space and has low computational complexity. NSGA-II finds a Pareto optimal front of solution space due to the presence of two objectives in the above problem: minimizing both power consumption and MLU. In this genetic algorithm (GA), the solution is encoded through a chromosome, which is created from a number of genes equal to the number of network links. We introduce several more GA custom operators of crossover and mutation operators. These latter operators should further enhance NSGA-II basic operators and make the search in the Pareto-optimal solution space more efficient.

### **GA Solution Encoding, and Scheme Overview**

Each chromosome (population member) has  $L$  genes where  $L$  is the number of links. Each gene is represented as an integer value within a range of  $[1, 65535]$ , which is part of the link weights vector  $w = (w_1, w_2, \dots, w_L)$ , where  $w_i \in [1, 65535]$  for each link  $i = 1, 2, \dots, L$  (Ericsson, Resende, & Pardalos, 2002). Each gene represents a link weight and is restricted to an integer value in the range of 1 to 65,535 (Fortz & Thorup, 2000). Links that are not used to carry traffic are selected to be put in sleep mode, by using a link weight value of 65,535. However, the latter selection mechanism is up to the

network operator, as another scheme or range of link weights can be used to enforce putting links into sleep mode. Each chromosome has two recognizable fitness functions in the NSGA-II algorithm that represent equation 1 and equation 2, respectively. Figure 2 shows the encoding of each chromosome and how each gene represents an integer value that corresponds to a link weight. The higher the values of a link weight the lower probability that the network operator will select this link for carrying traffic. The network operator can tune the integer link weights of the links, in order to compute shortest paths for carrying the network traffic (Rexford, 2006). Load balance uses Open Shortest Path First (OSPF) link weight management. Using OSPF, the network operator assigns a weight to each link, and shortest paths from each router (node) to each destination node are computed using the weighted cost function of the links (Fortz & Thorup, 2000).

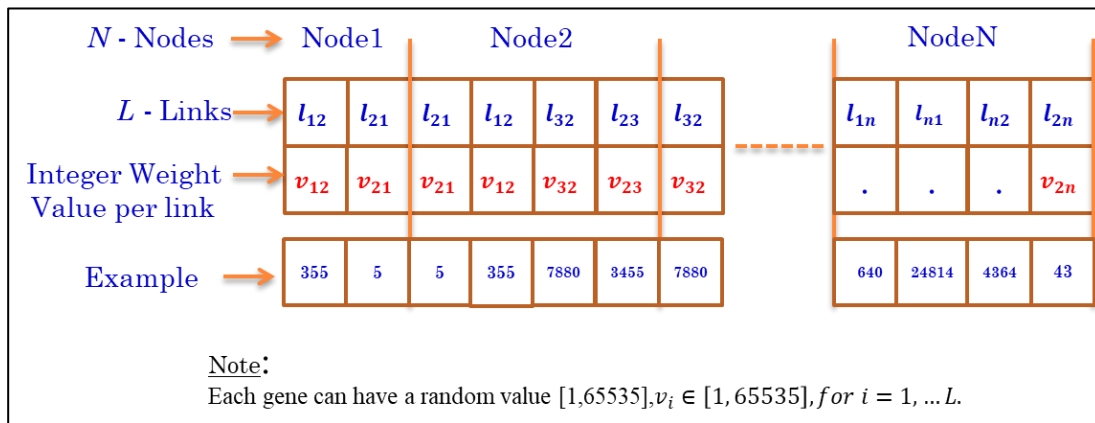


Figure 2 – Network Links Mapping to Chromosome Encoding.

Each node is connected to other adjacent nodes based on the given network configuration, and not necessarily to all other nodes. Figure 2 shows an arbitrary example of nodes connectivity to other links. For a given chromosome, each fitness function is evaluated numerically and outputs a power value for the first fitness function

and a numerical output of MLU for the second fitness function. When we use MOO, we supply a function that takes several inputs: a chromosome, network graph  $G$ , network demands  $D$ , and links' weights, and outputs two numeric values corresponding to the two objectives. Link utilization is measured when we route our traffic matrix on top of the topology with the updated link weights. Thus, the resulting chromosome with the optimal solution includes different values of its genes that eventually maps to a particular traffic routing.

### An Illustrative Example of Network Energy and MLU MOO Algorithm

Figure 3 illustrates the basic concept of link weight optimization for both minimizing power consumption and achieving load balance. We use a small example of network topology in Figure 3 assuming link capacities of all links are at 100%, and that MLU must be below 50% ( $\alpha$ ) as an example. We show a weighted directed graph  $G(N, L)$  where we have 5 nodes ( $|N| = 5$ ), and 8 links ( $|L| = 8$ ), with the indicated link utilization and link weight settings for each link. These link weights were selected for the sake of describing the optimization of link weights in order to save energy, while not exceeding the MLU limit.

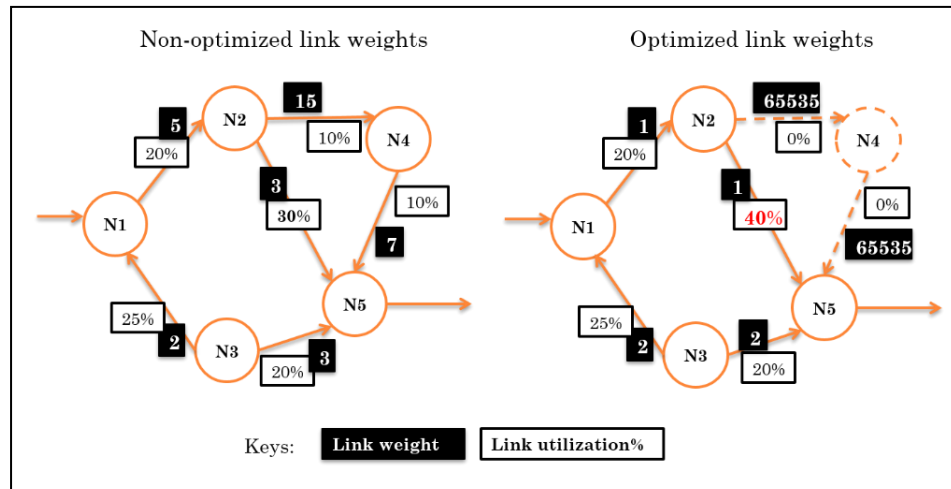


Figure 3 - Illustrative example for network topology link weights optimization.

The path of N1-N5 can go on multiple traffic paths, through N1-N2-N5, N1-N3-N5, or N1-N2-N4-N5. Clearly, we can shut down links N2-N4 and N4-N5 without causing the network topology to lose full connectivity. We first consider the case where a set of link weights are not optimized as in the left side of Figure 3 where the maximum link utilization of 30% is currently found at link N2-N5. The case where the link weights are optimized is depicted on the right side of Figure 3. Whenever we want to minimize the power consumption of this network topology, we select shortest paths and we increase their utilization assuming that we do not exceed its allowed MLU. Therefore, the algorithm does not select links N2-N4 or N4-N5, which are the least utilized links. The MOGA algorithm uses a special link weight of 65,535 to represent that the link is not utilized and it should be put to sleep. The MLU is increased, since N2-N4 and N4-N5 are put to sleep, and their traffic demands is routed along the other path of N2-N5. Evaluating the maximum link utilization function should yield 40%, which is found on link N2-N5. The power consumption is calculated based on equation 1, as two links are put to sleep and thus reduce the value of the first fitness function of power consumption.

The example above shows that optimization of link weights has allowed two links to go to sleep by selecting shortest paths for carrying the traffic, while not exceeding the allowable MLU of the network. Calculating the objective functions is the most expensive process computationally; since we have to repeat it for every GA generation (each chromosome has  $L$  number of genes or links). For every GA generation we have to evaluate the two objectives fitness functions for the  $P$  population size. GA stops when the best solution has not improved in the last *gamax\_num* number of generations.

In this research, our approaches rely on sampling traffic changes and metrics in a continuous fashion. This approach extends Zhang et al. (2010) combined method for saving power and at the same time control the link load. They use link rate control by examining rates (measured in Giga Bits per Second or Gbps) of links according to their traffic condition, as well as considering sleeping of links, routers and switches. Running our MOO uses adaptive strategies that optimize two objectives of minimizing power consumption and minimizes MLU. Our energy-aware network configuration uses shortest paths and our network load-balance tracks changes in traffic flows.

### **Traffic Engineering Schemes**

All of the developed approaches in this research are based on MOO. Network operators employ TE schemes in order to perform load-balancing. All approaches rely first on certain energy aware TE schemes that are based on sorting criteria or rules for the nodes and links (Amaldi et al., 2013; Chiaraviglio et al., 2012). The sorting criteria for nodes that we consider for this work are shown in Table 2. They include random (R) and Most-Power (MP) for nodes, Least-Flow (LF) for both nodes and links, and Least-Link (LL) for links.

Table 2 - *Criteria for link/node sleeping.*

<b>Sorting Criteria</b>	<b>Applies to</b>	<b>Description</b>
Least-Flow ( <i>LF</i> )	Nodes and Links	Consider smallest amount of traffic
Random ( <i>R</i> )	Nodes and Links	Random order
Most-Power ( <i>MP</i> )	Nodes	Consider highest power consumption
Least-Link ( <i>LL</i> )	Links	Sort link weight values monotonically

In the following sections, we describe the three approaches that use the sorting criteria listed in Table 2.

### Dual Approach 1 (MOGA-I)

In the first approach, the algorithm concurrently considers both the link sleeping scheme, and the link weights. It uses random (R) criteria for selecting the value of each link, which indicates Link Sleeping State (LSS), whether it is active or asleep. In this approach the network measurement and initial or current link weights are not considered, and the MOGA-I algorithm finds optimal solutions driven by MOO, in order to minimize both fitness functions (1) and (2). The link weights are obtained by running MOGA-I using the gene and chromosome encoding as shown before. We use two kinds of chromosomes as depicted in Figure 4. One chromosome uses binary encoding (0,1) for each link  $l$  from node  $i$  to node  $j$ , so it includes  $L$  genes where every gene has LSS value. The second chromosome uses Integer Weights Values (IWV) per link as shown before. Each gene can have a random value  $[1,65535]$   $v_i \in [1,65535]$ , for  $i = 1, \dots, L$ .

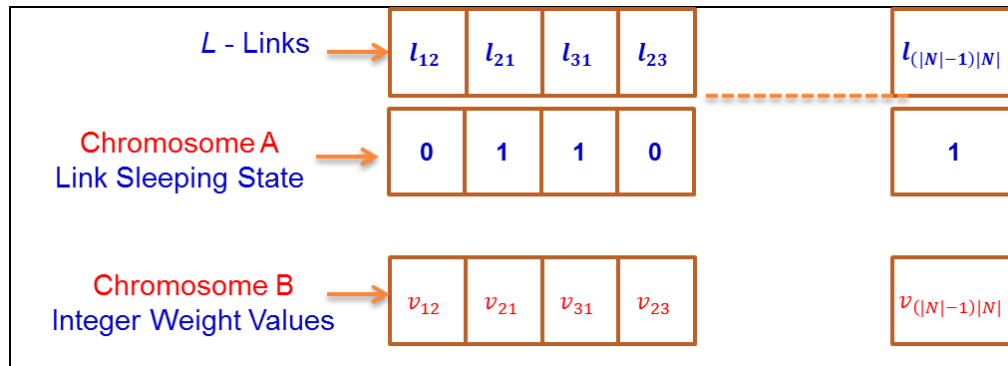


Figure 4 - Dual Chromosome Approach.

The MOGA-I algorithm generates both chromosomes randomly, and it uses GA operators such as crossover and mutation in order to get optimal solutions. Our network

topology system captures the network topology, injects the traffic flow, and implements the optimization algorithm mechanism. Fitness functions are calculated every time based on the chromosome values, and based on running the network topology system in order to decide on the final load balancing and links utilization in order to calculate MLU. This network topology system uses OSPF routing and uses Equal Cost Multiple Path (ECMP) as part of the shortest paths routing. Decision variables  $x_{ij}$  are obtained by MOGA chromosome A, while  $y_m$  is generated by  $x_{ij}$  if all links that are flowing into and out of a certain node  $m$  are all inactive or asleep. Power consumption (PC) fitness function is easily calculated based on  $x_{ij}$  and  $y_m$ , and by considering the power information of associated nodes and links. Consequently, new network flows  $f_{ij}^{sd}$  and  $f_{ij}$  are obtained as the output of the network system along with the fitness functions numerical values.

### **Delta Weight Approach 2 (MOGA-II)**

In this approach, we use the values of the link weights as configured in the current network configuration as our initial link weights set of values. In order to minimize oscillations in the network when applying new link weights and configuring new traffic flow matrix, we rely on changing link weights gradually based on relatively small difference in each link weight value denoted as  $\Delta w$  (delta weight). We must make sure that the absolute value of  $\Delta w$  is carefully selected as it has to be big enough for the network operator to decide to change the traffic flow through that link. On the other hand,  $\Delta w$  should not be too big in order to prevent oscillations in the network.

This method compared to the dual approach discussed earlier, uses one chromosome B that uses the IWV for each gene's value, but uses different weight values range per link. In this method, chromosome A is pre-assigned before entering the

MOGA-II algorithm (Multi-Objective Genetic Algorithm for this second approach), based on one of the rules in Table 2. A specific TE scheme (Table 2) is used such as LF, which considers least loaded links in order to evaluate the links LSS values. The least loaded links, which have traffic utilization under a specific limit, become candidates for being put to sleep. The MOGA-II algorithm uses a two-step solution, where in first step the LSS values are pre calculated and the decision on those nodes and links that are put asleep is already made. Pre calculation of LSS values uses least-flow (LF) criteria for selecting the value of each link. The MOGA-II algorithm is described in Figure 5.

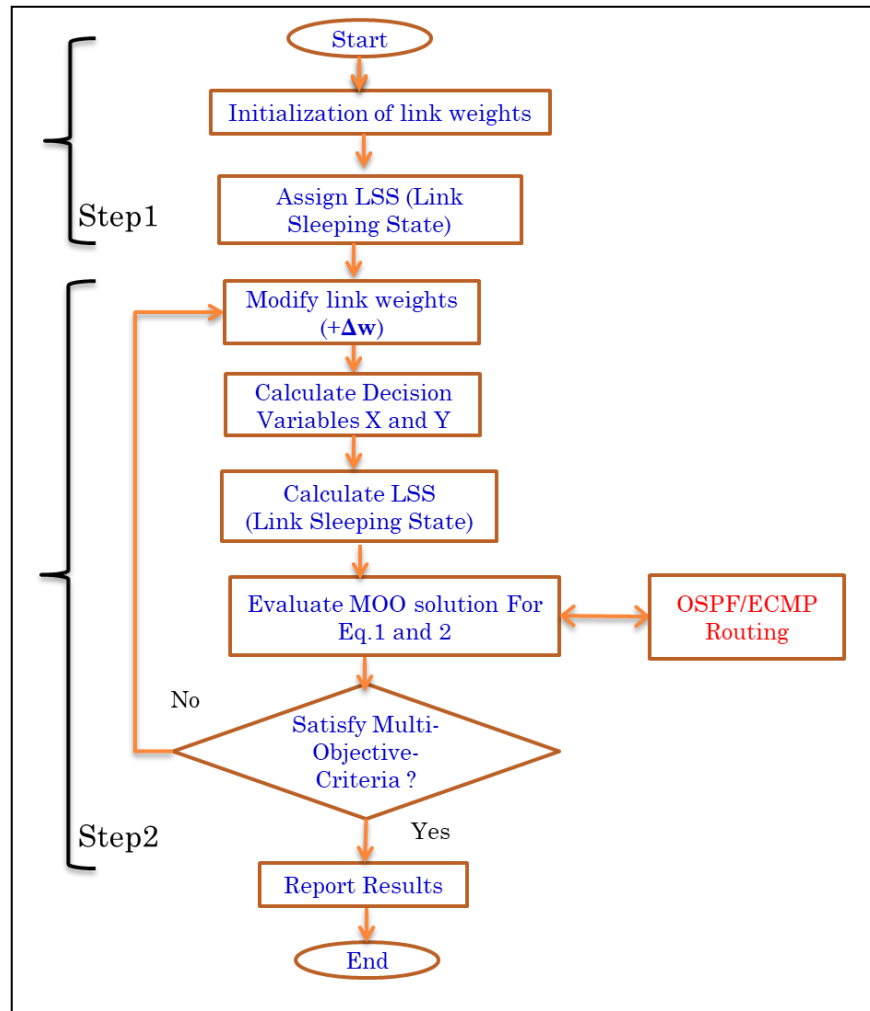


Figure 5 - MOGA-II Two Step Algorithm.



In the second step, the MOGA-II algorithm runs in order to find optimal solutions for minimizing objectives from both equations 1 and 2 of PC and MLU, respectively. This MOGA-II algorithm uses chromosome B, but it uses different values for the link weights. So IWV is extracted historically based on previous or initial link weight value, and it is later modified by an incremental value chosen from a limited range of values as a function of  $\Delta w$ . Assume  $w_{ij}^l$  is the initial weight value for the link  $l$  from node  $i$  to node  $j$ . The differential weight value  $\Delta w$  for this link is what we need in order to make the network operator change traffic through this link, either by reducing or by increasing its total link weight. We use  $\Delta w_{ij}$  to denote  $\Delta w$  for the link  $l$  from node  $i$  to node  $j$  at time  $T$ . Eventually, MOGA-II should select weight value  $w_{ij}$  for each gene from a value range described below:

$$w_{ij} \in [(w_{ij}^l - \Delta w_{ij}), (w_{ij}^l + \Delta w_{ij})] \quad \forall (i, j) \in L \quad (6)$$

The value of  $\Delta w_{ij}$  is carefully selected based on inputs from the network operator and based on the OSPF/ECMP algorithm. Those inputs specify the value needed for deviating traffic from a certain link. Furthermore, modification of the link weights after we evaluate MOO (picking the “No” branch of “Satisfy Multi-Objective Criteria”) is done by increasing the weight of least loaded links (LF) by  $\Delta w_{ij}$  (make them sleep), or by increasing the weight of other links by  $\Delta w_{ij}$  (deviate traffic flow). The parameter of  $\Delta w_{ij}$  is re-computed and the updated new link weight is represented as follows:

$$w_{ij} = w_{ij}^c + \Delta w_{ij} \quad \forall (i, j) \in L \quad (7)$$

$$\Delta w_{ij} = (w_{max} - w_{ij}^c) \quad \forall (i, j) \in L \text{ if } x_{ij} = 0 \quad (8)$$

Note that  $w_{ij}^c$  is the current link weight value for the link  $l$  from node  $i$  to node  $j$ , and  $w_{max}$  denotes the maximum link weight value in the current network configuration as evaluated by MOO. In the problem formulation it was indicated that if  $x_{ij}$  is 1 then the link is active, otherwise if 0 then it is inactive or asleep. If the new link is still active, then based on equation 7 its weight  $w_{ij}$  is updated by adding  $\Delta w_{ij}$  to the current link weight  $w_{ij}^c$ . Otherwise if the new link is asleep then based on equation 8 its weight of  $w_{ij}$  is set to  $w_{max}$  and the link  $l$  from node  $i$  to node  $j$  remains at sleep state. As noted, our main goal is to switch off the maximum number of links for energy saving and then spread the traffic as evenly as possible on all remaining active links.

This approach of using  $\Delta w$  along with the MOGA algorithm can reduce the computation time of the algorithm since it uses a smaller search space. The only drawback of this  $\Delta w$  approach is that it finds fewer optimal solutions due to the smaller search space than that used in the random dual approach 1 (MOGA-I) above.

### **Hybrid Approach 3 (Hybrid MOGA-III)**

This approach combines approach 1 and 2, where random approach is executed first, for number of generations. The best solution is then chosen to be used for the delta weight approach that runs for further number of generations. The main advantage of this hybrid or mixed approach is that it combines the best of approach 1 and 2, since approach 1 yields better solutions while lacking stability and has more oscillations. On the other hand, approach 2 has less oscillation in network traffic and more stable, but has a smaller search space and yield less good solutions. The idea is that the algorithm runs for a number of generations  $N1$  using approach 1, and then after  $N1$  generations, it switches to

approach 2. After  $N1$  generations, the best solution is tracked and is promoted to be used for approach 2. The algorithm chooses the best solution from the Pareto Front solutions.

### **Common Issues in All Three Approaches**

All approaches yield new link weight values, which are flooded in the network using control messages. The routers then re-compute the shortest paths (OSPF) and update their forwarding routing tables; this may take a few seconds until all routers stabilize their shortest paths (Moulierac & Phan, 2014). We addressed this network disruption and instability of link weights in the discussion of the approaches below. The more excessive the link weight changes the more network noise and oscillations occur. Furthermore, the runtime of algorithms is targeted to run in less than a minute, and this is addressed accordingly in the implementation and results phase.

In all three approaches, and after each execution of our MOO, we keep track of the “top” node which is selected from those nodes that were turned off so far. This “top” node yields the highest traffic capacity considering all links that are connected to it. If the MLU increases above a specific threshold  $\alpha$ , then we turn this “top” node back on.

### **Adaptive GA Genetic Operators**

All approaches use GA or MOGA to search for good solutions using mutation and crossover operators are described below. In this work, the GA operators are adaptive, as they depend on the network configuration inputs of topology  $G = (N, L)$ , traffic matrix  $D$  and the initial set of current link weights. The fitness functions depend on those inputs and can have a different solution space every time those inputs change.

Mutation and crossover operators are aware to LSS, while taking one solution candidate (chromosome) and altering the values (IWV) of one or more genes in the

chromosomes (Francois et al., 2014). Therefore, both operators were customized to fit the problem that we have, by considering both LSS and IWV values for every network configuration.  $G$ ,  $D$ , and initial link weights represent each network configuration.

#### *GA Crossover Operator*

In GA, a crossover operator takes two chromosomes (i.e. two solutions) in the current population and swaps their genes (i.e. link weights) with each other in order to generate two new offspring chromosomes. We use the most common crossover and mutation operators that are the two-segment crossover, and random gene mutation, respectively. In addition, a new crossover operator compares two parent chromosomes, and performs a multiplication function (AND) between the LSS value of each link that is associated with two different parents gene (i.e. link weight). This crossover operator may urge link sleeping to multiply through the population, since one parent chromosome can alter its gene with its correspondent gene in the other parent if it represents a sleeping link. Figure 6 shows an example of parent chromosomes where each parent is composed of links which represent its genes. The LSS values for the two parents are shown, and if the link is sleeping then the value of LSS is “0”, otherwise the link is active and the value of LSS is “1”. This customized crossover operator is designed so that one parent chromosome can replace its gene with the other parent chromosome corresponding gene if it represents a sleeping link. The resulting child as shown in Figure 6 has the value of its gene’s LSS as the AND function between the two parents. The arrows between the two parent chromosomes show the direction of the change when the LSS change occurs. This new LSS aware crossover operator allows those link weights that hold the attribute of link sleeping to multiply through the population.

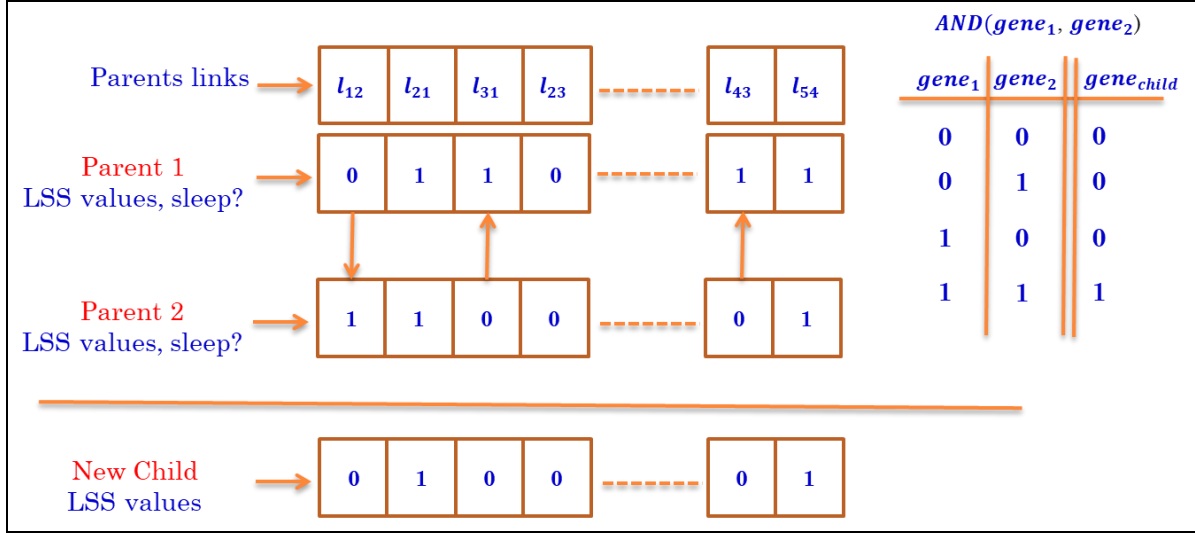


Figure 6 - LSS Aware Crossover Operator

### GA Mutation Operator

Another GA operator that we have customized is the mutation operator, which modifies one or more genes (i.e. link weights) in the chromosome. We associate the mutation probability of each link (i.e. gene) directly with the link utilization and MLU in the network, such as  $P_{ij}^{mutation} = \frac{u_{ij}}{MLU}$  as described in the context of the work in (Francois et al., 2014). Whenever the link is more utilized, its mutation probability becomes higher. This way a link that has high utilization will have its link weights increased, and thus less likely to be selected by the network traffic routing. Anytime we generate a chromosome via mutation, the mutation can be done either by the new customized mutation method or the random one, but not by a mixture of both for the same chromosome. If the new customized mutation method is used, each gene is mutated with the probability linked to the utilization of the link that the gene represents, and the link weight changes according to the function in equation 9 below:

$$w_{ij} = \begin{cases} w_{ij}^c + \Delta w_{ij} & \forall (i, j) \in L, \text{ if } u_{low} > u_{ij} \text{ or } u_{ij} > u_{high} \\ w_{ij}^c - \Delta w_{ij} & \forall (i, j) \in L, \text{ if } u_{low} < u_{ij} < u_{high} \end{cases} \quad (9)$$

In equation 9, we compare the link utilization value to pre-established limits for maximum link utilization  $u_{high}$  and minimum link utilization  $u_{low}$ . Highly utilized links should have their link weights increased in order to reduce MLU, so that there is more chance that traffic goes away from them. Low utilized links should have their link weights increased as well so they go to sleep. Medium utilized links should have their link weights decreased in order to take more traffic. Thus, this method adaptively tunes mutation rate according to the specific inputs from the network.

### **Network Instability and Delay Parameters**

In order to address network instability after changing the link weights, and to better understand the effect of delay on runtime, several experiments were carried out. These experiments prove reliability or consistency of measurements of link load and power consumption across various simulations conditions.

Two types of methods are used in order to isolate and reduce the traffic flow instability after changing the link weights. First, we apply a sequential delay whenever we change the link weights for an individual node that belongs to the network. Each individual node has its links' weights altered in a sequential fashion, one after the other until we go through all nodes that belong to the network topology. Second, we apply a wait interval delay after we alter the weights of all node's links. Eventually, in every iteration we apply two kinds of delay parameters, *sequential\_delay* parameter and *wait\_interval* delay parameter. For both network topologies that we tested, a similar set of granularity in setting delay numbers was used.

Prior work of Shaikh et al. (2000) described a failure due to network instability caused by congestion, so we chose to analyze the traffic load as a measurement method

for validating our work. A formula for describing the traffic flow differential measurement is shown below:

$$\delta_{instability\_k} = \frac{MLU(t+1) - MLU(t)}{MLU(t)} \quad (10)$$

The above formula in equation 10 describes the instability factor in each traffic flow measurement,  $MLU(t)$  is the traffic flow maximum link utilization value returned at time  $t$ , and after sleeping 1sec we return the value of  $MLU(t + 1)$ . We calculate the difference in traffic flow MLU and divide it by  $MLU(t)$ . This gives an indication of how much traffic flow change has occurred. At each iteration  $k$ , we measure  $\delta_{instability\_k}$ , and at the end of the simulation, we calculate the average based on all  $n$  iterations. The next formula described in equation 11, shows how we calculate the average for the entire simulation:

$$average\_ \delta_{instability} = \frac{\sum_{k=1}^n \delta_{instability\_k}}{n} \quad (11)$$

We calculate  $average\_ \delta_{instability}$  after each simulation, and for each simulation, we vary mainly the delay parameters between each iteration, and internally in the same iteration while changing each link weight as described in the next sections.

### *Sequential Delay Method*

After running few experiments, we noticed that traffic flow disruption is highly affected by each link weight change, and it depends on LF TE scheme and the randomness of GA algorithm. Figure 7 shows the flow diagram and describes how we change the link weights together with the *sequential\_delay* parameter. Each link that

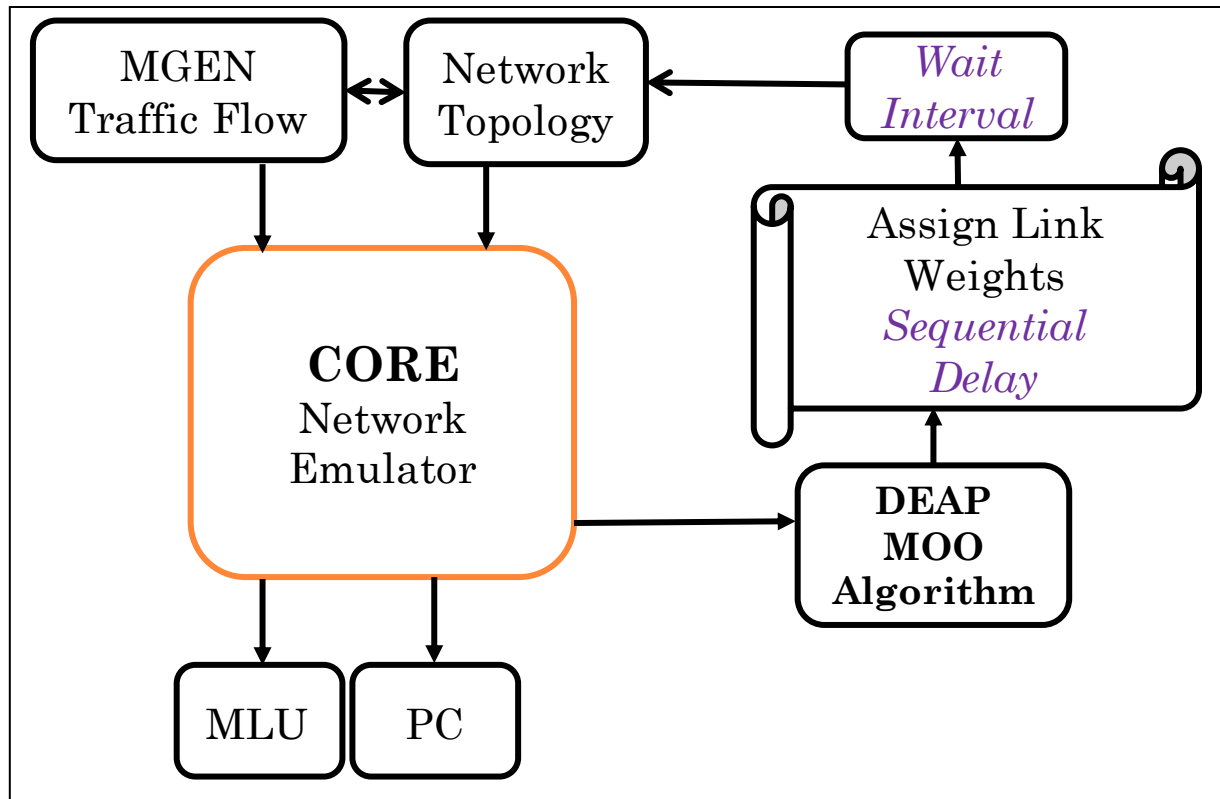


Figure 7 - Diagram of CORE and Algorithm for Setting Instability Delay Parameters

belongs to a specific node has its weight altered based on our MOO algorithm. After each node's links weight are set, then we apply a *sequential\_delay* that allows the traffic to settle down and thus reduce network oscillations. We apply new link weights for all nodes' links along with adding *sequential\_delay* after each node's links are assigned. Consequently, we apply additional *wait\_interval* delay as shown in Figure 7, and after that, we measure the resulting MLU and PC. In the next section, we discuss the *wait\_interval* delay parameter.

#### Delay Wait Interval Method

In the second step, we can also apply a global delay *wait\_interval* after all link weights have been changed in each iteration, and between each iteration and the next



iteration, this *wait\_interval* delay is implemented. Applying a *wait\_interval* delay after all weights of all links have been altered allows CORE emulator to settle down all links together and gives rise to better stability in the network flow.

The network system and methodology is shown in Figure 7 above, where network topology and traffic flow using MGEN is fed into CORE network emulator. We measure the MLU and PC in each iteration and feed it back to MOO algorithm, which is implemented using DEAP. We describe two mechanisms for stabilizing the traffic flow, one mechanism is to include a *sequential\_delay* after each link weight assignment after MOO algorithm is run, and second mechanism is to include a *wait\_interval* after finishing each GA iteration that alters settings of links' weights of all nodes. For each node that belongs to the network topology, we apply a *sequential\_delay* after all of its links' weights are applied.

We start the CORE simulation by using default link weights, and once the traffic is established we start running the MOO algorithm that yields new link weights for all links in network topology. In every iteration, we collect the resulting MLU and PC and after several iterations, DEAP keeps track of HOF (Hall of Fame) best Pareto optimal front solution so far in the algorithm. Those resulting MLU and PC are also used as an input for DEAP mutation and crossover functions as described earlier in our methodology. As shown in Figure 8, adding the delay is done in two parts, one part that applies a *sequential\_delay* ( $d_{seq}$ ) after each change of link weights for each link of the node, and a second part that applies *wait\_interval* ( $d_{wait\_interval}$ ) after all new link weights of all of the nodes are applied.

○ **Input:** New optimized link weights per each node and its links  
 ○ **Output:** Updated network traffic link load, power consumption and instability measure

```

1. begin
2.  $i = 1$ ;
3.  $global\_Iterations = MOO.sumOfDEAP\_Iterations$ ;
4.  $d_{seq} = sequential\_delay$ ;
5.  $d_{wait\_interval} = wait\_interval$ ;
6. Initialize link weights (Default);
7. while  $i < |global\_iterations|$  do
8.   RunCoreNetwork( $D, N, L$ );
9.    $IWV(L) = RunDEAP\_MOO\_Algoritm$ ;
10.  for each  $n \in N$  do
11.    for each  $l \in L(n)$  do
12.       $IWV[l] \leftarrow value$ ;
13.      Apply_link_weight( $IWV[l]$ )
14.    sleep( $d_{seq}$ );
15.  sleep( $d_{wait\_interval}$ )
16.  Perform Measurement on MLU and PC
17.  Measure Instability using MLU values
18.   $i = i + 1$ 
19. return multi – objective opimal solution of both MLU and PC
20. return average_ $\delta_{instability}$ 
21. end
  
```

Figure 8 - Pseudo Code Showing *wait\_interval* and *sequential\_delay* Algorithm

The instability algorithm described in Figure 8 keeps track of each iteration and applies the two types of delay parameters. IWV values as described in Chapter 3 are determined by MOO algorithm, and for each node and its links, we apply a *sequential\_delay* and a *wait\_interval* delay as shown in the pseudo code in Figure 8. Adding the *wait\_interval* delay improves further the instability measurement and allows the network to stabilize more. At the end of the simulation, DEAP gives the multi-objective optimal solution for both MLU and PC from the Pareto optimal front as discussed earlier in our methodology.

## Resource Requirements

In order to test our methodology in real time and using a flow based traffic we used a network emulator that can resemble a real computer network running on actual routers and links. Common Open Research Emulator (CORE) is a computer network emulation tool from Naval Research Laboratory (NRL), and it was used to emulate the network and capture the network topology (Ahrenholz, Danilov, Henderson, & Kim, 2008).

We used CORE for calculating the MLU fitness function, which simulates a computer network topology. Widely used SNDLib provides real network topologies like Abilene and Polska (Orlowski, Wessäly, Pióro, & Tomaszewski, 2010). MGEN is both a command line and GUI traffic generator that was developed by US Naval Research Laboratory. We used MGEN tool to generate real network traffic flows that can be fed to CORE. We calculate the second fitness function of PC in parallel by a Python utility, which extracts and reflects the values of the network flow. Both fitness functions are fed back to a Python program that implements our MOGA algorithm and figures out the best solution.

MGEN runs on Linux platform, and provides programs for sourcing or sinking real-time multicast/unicast IP traffic flows. The multi-generator (MGEN) can generate real-time traffic patterns to load the network with Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) IP traffic. The MGEN tool transmits and receives time-stamped, sequence numbered packets. The analyses of the log files can be performed to assess network or network component ability to support the given traffic load in terms of throughput, packet loss, delay, and jitter.

Similar experiments and test-cases are used such as in Vasić et al. (2011) and Zhang et al. (2010), who used Abilene and GEANT network topologies and traffic matrices (Tune, Roughan, Haddadi, & Bonaventure, 2013). After we import the network topology from SNDlib, we run CORE emulation, that utilizes OSPF routing protocol using Quagga software, which is software package for TCP/IP networks (Jakma & Lamparter, 2014). The Quagga Routing Suite provides implementation of several common routing protocols including OSPF (Nascimento, Rothenberg, Salvador, & Magalhães, 2010). We configured the Quagga software in order to run with ECMP, and CORE uses Quagga/Zebra software for managing the OSPF and routing parameters. Furthermore, by using Quagga shell “*vttysh*” we can configure the OSPF link weights on the fly while CORE network emulation process is running

The above network simulation tools capture the objective functions that are fed to a software program written in Python. Latter program uses Distributed Evolutionary Algorithms in Python (DEAP) for the multi-objective optimization. DEAP is a novel evolutionary computation framework that is used for Genetic Algorithms (GA), and Multi-Objective Optimization such as NSGA-II, and Strength Pareto Evolutionary Algorithm 2 (SPEA2).

### **Summary of Methodology**

Energy aware networking is an active area of research and it is gaining momentum, as numerous researchers are still investigating it, and are developing new optimization techniques and methods for reducing power consumption in IP networks. Hence, saving energy in IP networks is well in demand, and any new technique for realizing it is important. The research work that was described here results in a method

for reducing IP network power consumption while considering network traffic that is close to real-time traffic as much as possible, and it takes less than few minutes for altering the traffic paths and realizing the energy reduction in the network.

Our main research activity developed MOO techniques for finding good solutions that satisfy the two said objectives of PC and MLU. Achieving better results relies on the two types of data sets that we used, which are the real network topology and traffic demands that should resemble real internet traffic. The hybrid approach MOGA solutions' diversity is addressed by running random approach, while intensification of search is achieved by using delta weight approach in the consequent algorithm iterations.

Another major part of our study was researching the instability of network traffic whenever we alter the link weights of the nodes. We conducted a research for studying the effect of varying the *wait\_interval* delay and *sequential\_delay* on the traffic instability and measurement. It improved the traffic measurement by tuning those delay parameters.

In the future, techniques for sharing the selection algorithms of SPEA2 and NSGA-II can be used, by developing the same baseline simulations that were carried out in this research. Nevertheless, solving the problem in this study in the future can use latest algorithms of MOO that may give better Pareto set of solutions that are evenly distributed along the Pareto front (Jain & Deb, 2013). Evolving the approach in this study in order to support the tremendous increase in IP networks with larger topologies and traffic matrices is another area of future research.

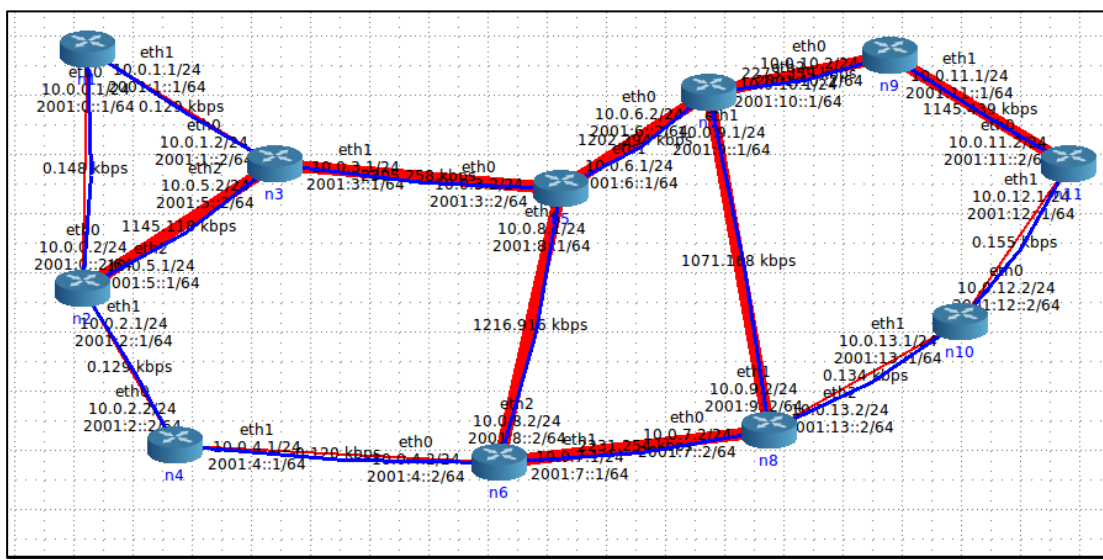
## Chapter 4 - Results

### Introduction

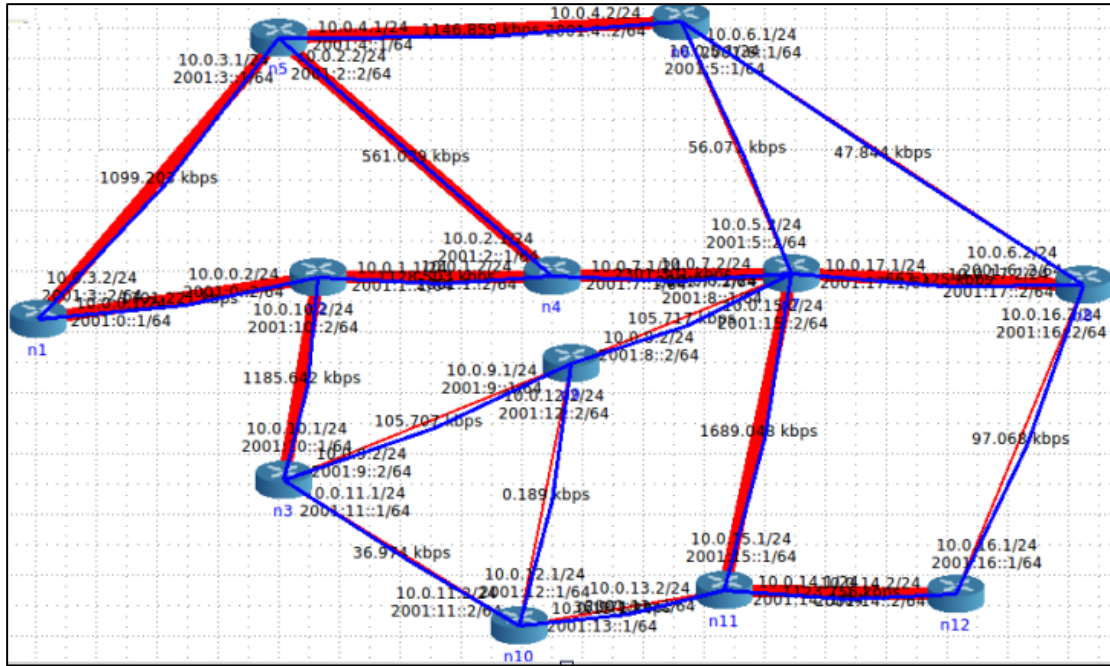
The goal of this dissertation was to develop adaptive strategies for a multi-objective genetic algorithm such that both objectives of reduced network power consumption and load balance are achieved. In order to achieve this goal a new approach was introduced and implemented using network real time traffic that was generated by MGEN. This research investigated the use of MOO in order to find good solutions for reducing PC while maintaining network MLU below certain allowed threshold. Evaluation of new methods for reducing and measuring network traffic instability is also included in this chapter. In the following sections, we show the results of running the three approaches discussed in Chapter 3 based on two types of network topologies; Abilene and Polska that are shown in Figure 9. Later in this chapter, we present the results of instability study that we discussed in Chapter 3.

### Network Traffic and Data Sets Description

Performance of the three various approaches discussed in previous methodology Chapter 3 is first evaluated on network topology of Abilene Orłowski et al. (2010) which has been used for Academic research since 2004. This topology is shown in Figure 9 (a) and it includes 11 nodes and 14 links. The power metrics used in this study are based on the same information used for GreenTE algorithm in the work of Zhang et al. (2010), and based on Cisco line cards information. We use 0.6 kW for each link and 10 kW for each node. For the sake of simplicity, we assume that all network links have the same traffic capacity. Both links and nodes are considered when calculating the total power consumption of the network consisting of links and nodes. We picked three network



(a) Abilene



(b) Polska

Figure 9 – (a) Abilene Topology, (b) Polska Topology

traffic data sets from Abilene recorded data, which correspond to 5<sup>th</sup> of June, 5<sup>th</sup> of August, and 5<sup>th</sup> of September. The data sets depict the total traffic generated in each of

those days per minute of time. We then translate the data sets into MGEN data format using Poisson distribution.

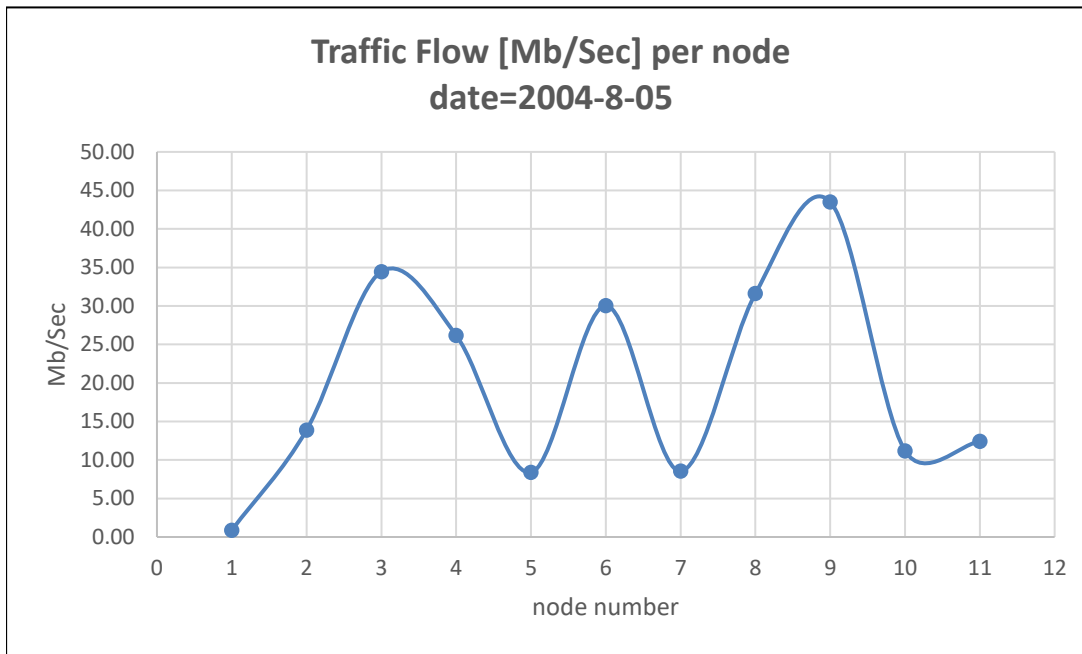
#### *Network Data Sets and Usage*

Figure 10 (a) describes the actual raw data set that we used for Abilene, taken on 5<sup>th</sup> of August, 2004. The data sets were obtained from SDNlib and also verified the same from the work of (Tune et al., 2013). It shows how the arrangement of data per each node versus the rest of the nodes as a topology matrix. The raw data in each matrix element shows how much traffic is flowing from the left node indexed in row to the node in upper column index. Figure 10 (b) shows the amount of traffic in Mb/Sec that flows in each node taken as an example from August 5<sup>th</sup> of 2004 Abilene topology data sets. We translate this format of flow description to MGEN tool format, by specifying the flow



# time = 23-10-00											
# topology =											
# name = Abilene	2004										
# sampling = 1 in 100 packets											
# additional information = as measured using netflow											
# type = ingress/e router/pc traffic matrix											
# cite = Network / Yin Zhang Zihui Ge Albert Gr Matthew ACM/Use Berkeley CA USA 2005											
# units = Gbytes / second											
0.000549868	9.03E-06	2.97E-05	6.83E-06	7.18E-05	6.67E-06	3.20E-05	6.67E-06	0.000161	0	3.33E-06	
8.54E-06	0.002945	0.000537	0.000718	0.004144	0.001454	0.000113	0.002086	0.001281	4.09E-05	0.000571	
2.67E-05	0.002682	0.009015	0.00119	0.001009	0.007983	0.002308	0.006213	0.002539	0.00057	0.000924	
1.40E-05	0.001518	0.005631	0.001445	0.002835	0.00398	0.001423	0.003623	0.002013	0.00144	0.002262	
4.57E-05	0.002596	0.001139	0.000236	0.002101	0.000649	0.000232	0.00085	0.000374	1.34E-05	0.000149	
1.73E-05	0.001533	0.009319	0.00212	0.001321	0.003343	0.000507	0.005751	0.003457	0.000131	0.002526	
1.42E-05	0.000489	0.001493	0.000505	0.000513	0.002067	0.001491	0.000844	0.000242	0.000364	0.000525	
3.33E-06	0.002376	0.009396	0.002027	0.00234	0.003357	0.000887	0.002255	0.003932	0.002648	0.002386	
0.000152046	0.003044	0.003403	0.004415	0.000827	0.003619	0.002026	0.01	0.013222	0.000658	0.002145	
0	0.000109	1.44E-05	0.000478	0	2.64E-05	0	0.003629	8.39E-05	0.001381	0.005475	
0	0.001211	0.002505	0.000876	0.000895	0.000963	0.000151	0.002993	0.001862	0.000666	0.000292	

(a)



(b)

Figure 10 – (a) Abilene Traffic Data Set from 2004-08-05, (b) Abilene Traffic per Node

“from node  $i$  to node  $j$ ”, until all aggregate flows are combined and fed to CORE emulator. CORE emulator takes as input the converted MGEN traffic below, the network topology and configuration of OSPF, and Quagga routing flow.

The second network topology of Polska contains 12 nodes and 18 links as shown in Figure 9 (b) earlier. We could not find appropriate released data sets for this network topology, and as indicated in the research work of (Gianoli, 2014), we created similar traffic flow, and by using MGEN tool we developed a method to aggregate the traffic flow enough in order to create similar load balance and power consumption used in Abilene network topology. For power metrics, we used similar power consumption per each link and node as discussed in Abilene topology.

#### *Multi-objective DEAP Optimization Parameters Setup*

DEAP python package was used in order to optimize the fitness functions of MLU and PC. After installing the software, we had to tweak the main algorithm in order to accommodate our adaptive crossover and mutation algorithms as well as to select the best multi-objective solution and promote it from random approach to hybrid approach.

DEAP uses the idiom of Hall Of Fame (HOF) as the best Pareto optimal solution obtained after running for few iterations (Rainville, Fortin, Gardner, Parizeau, & Gagné, 2012). Table 3 shows the best values that we used in order to achieve reasonable savings in MLU and PC while considering the runtime increase in our algorithm. For example, NGEN indicates the number of generations used, and we have run our algorithm using either 25, 50, 60, 80 and 100 generations in each simulation. For CXPB (crossover probability) and MUTPB (mutation probability), we used constant values of 0.1 or 0.2, but we also used variable values based on adaptive LSS aware algorithm described in Chapter 3 of the methodology. APR\_CNT indicates the number of generation used for changing the approach from random to delta weight approach

Table 3 - *MOO Algorithm with DEAP program best parameters settings*

<b><i>NGEN</i></b>	25	50	60	80	100	
<b><i>CXPB</i></b>	0.1	0.2	<-- LSS aware and changes LSS of offspring			
<b><i>MUTPB</i></b>	0.2	var	<-- variable based on link utilization vs. total capacity			
<b><i>Lambda</i></b>	5	10				
<b><i>APR_CNT</i></b>	20	30	40	50	60	
$\Delta w$	1000	2500	5000	7500	10000	12500

In the case of obtaining two similar multi-objective solutions based on the Pareto optimal front, we tuned the algorithm so it selects the first item of HOF. Figure 11 shows a sample of 80 generations that we run on one of our experiments using Abilene network topology and shows all solutions represented as MLU in the Y-axis and PC in the X-axis. We drew a virtual Pareto optimal front that shows the closest solutions, and the red-circled solutions are those that HOF picked for us.

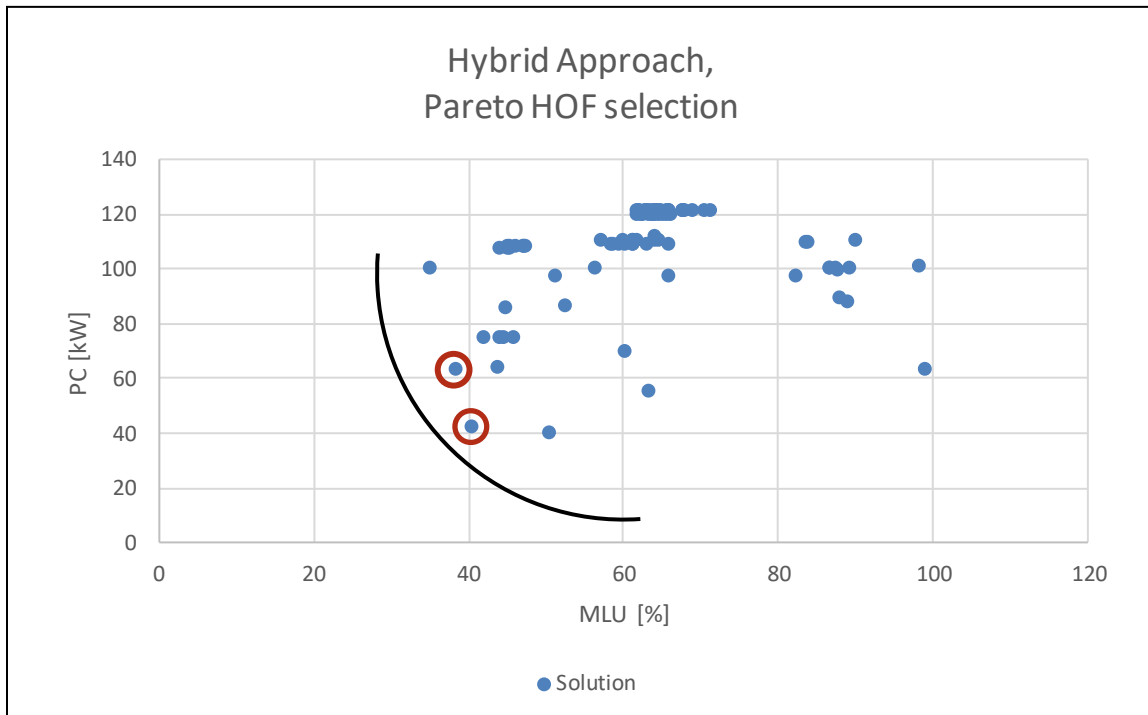


Figure 11 - Example of Pareto Front Solutions with Optimal Solutions

Another measure for validating and comparing the quality of DEAP's python solutions is to use a one-objective optimization function which combines both MLU and PC such that:  $WVAL = 0.75 * PC + 0.25 * NMLU$ . We used  $NMLU$  that is a normalized value of MLU, where we multiplied MLU by 100 in order to have comparable values to PC. We used this one-objective function as additional method of verification that HOF multi-objective function gives similar solutions.

The above formula favors solutions that have lower PC, than MLU. The latter formula enables us to evaluate numerous experiments due to the meta-heuristics nature of the simulation as random solutions are obtained, thus we run the simulations for 10 different times and then average the MLU and PC savings. This is similar to the work of (Francois et al., 2014), where they measured the average performance across 10 different

seeds in order to increase the confidence interval to 95% due to the non-deterministic behavior of the simulations.

Adjusting the traffic online requires specific parameters in our algorithm to be adapted and tuned in particular such as number of generations, mutation and crossover probability. Learning the network links states or Link Sleeping State (LSS), whether those are in active or sleep state was used adaptively in DEAP algorithm for setting the mutation and crossover operators.

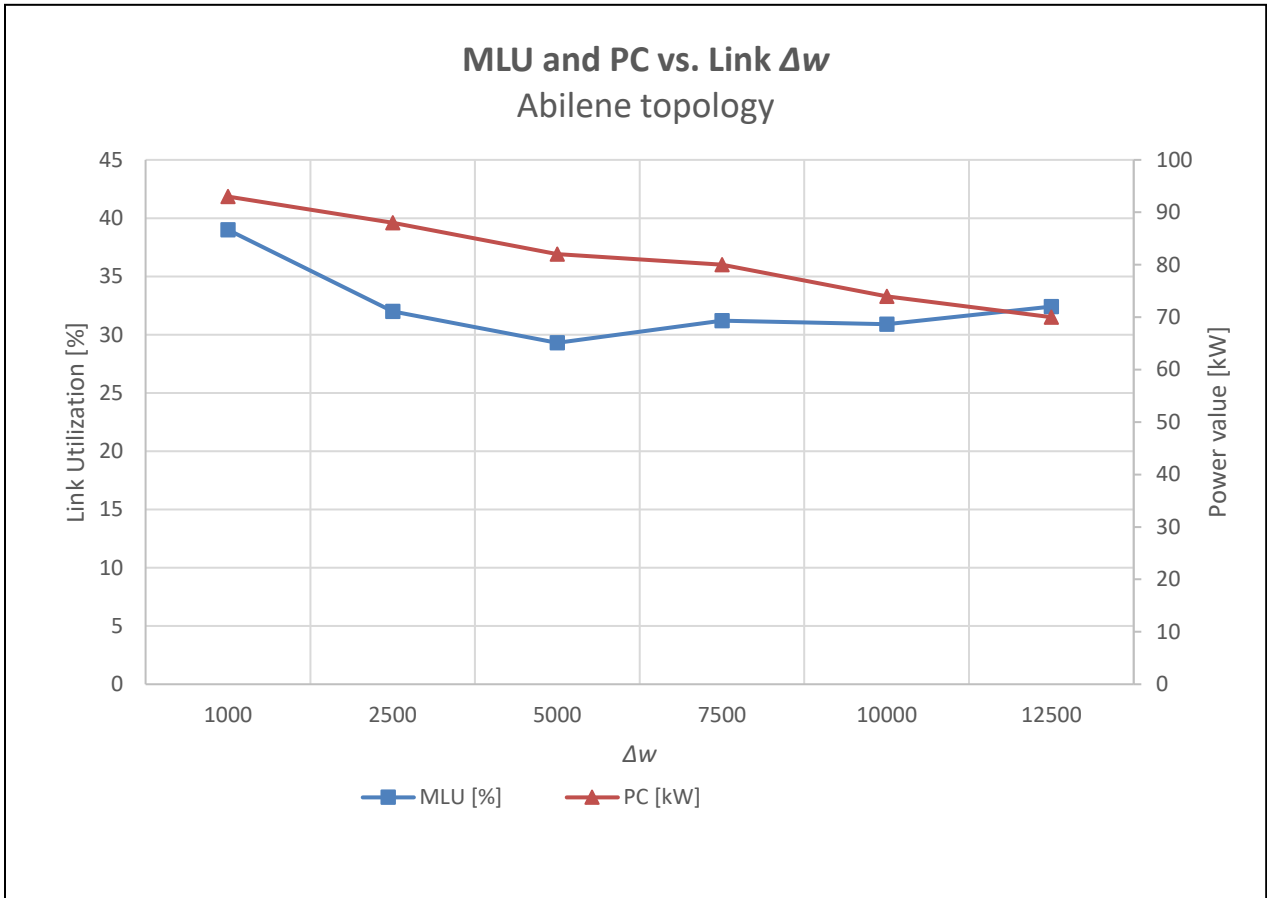
## **Simulation Results**

Extensive simulations were carried out in order to test the various approaches and to verify the potential for saving power while keeping MLU at acceptable level. We validated all approaches included a comparison of MLU and PC results which showed the best approach results. Additional instability simulations were performed, and the results shown in the subsequent sections compare the instability measurement with respect to various delay parameters.

### ***$\Delta w$ Simulations for the Delta Weight Approach***

For the delta weight approach, we use  $\Delta w$  in order to determine how much each link weight can be varied and obtain better solutions beyond the initial link weights that we started with. We varied the  $\Delta w$  in order to find the best value that yields good solutions for our multi-objective optimization algorithm. Figure 12 describes how MLU and PC vary as we change the  $\Delta w$  value from 1000 until 12500. We can see that the best MLU and PC multi-objective solutions are obtained when we increase  $\Delta w$  higher until we reach a value of 5000 where MLU is lowest. For PC, it keeps improving slightly as  $\Delta w$

increases beyond 5000 until 12500. Considering both objectives, we get that  $\Delta w$  of 5000 produces the best results.



*Figure 12 - MLU and PC vs. Link  $\Delta w$ , Abilene Topology*

For Polska topology, we get similar results when we vary the  $\Delta w$  value, with different levels of PC and MLU numbers, where we selected medium traffic for testing the  $\Delta w$  effect on MLU and PC fitness functions.

Figure 13 shows the results of best MLU and PC values that we obtained when we varied the  $\Delta w$  parameter, and as we can see from the graphs, we do get the best solutions around  $\Delta w$  of 5000, considering both objective functions.

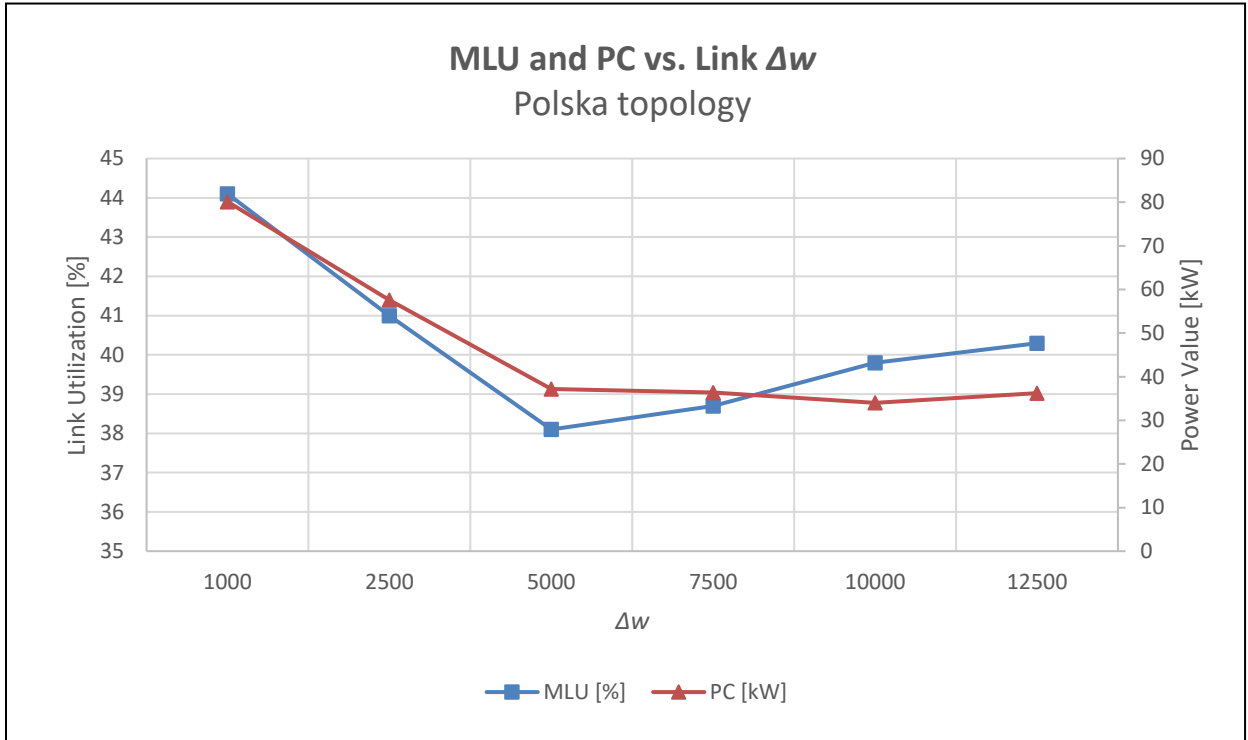
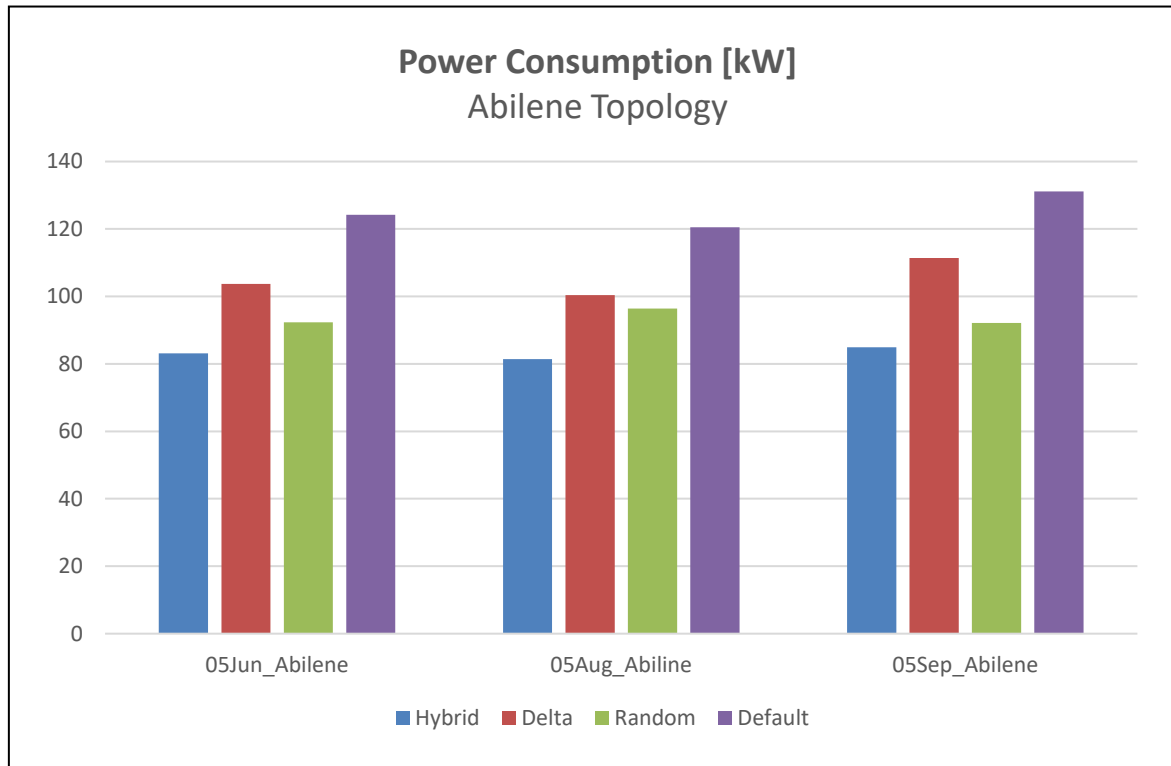


Figure 13 - MLU and PC vs. Link  $\Delta w$ , Polska Topology

### Abilene Network Topology Results

As indicated earlier, data sets were used as snapshots taken from SDNlib and then converted into MGEN traffic flow format (Orlowski et al., 2010).

Figure 14 and Figure 15 show the results of power consumption and MLU, respectively. We present 4 types of data results, where the first type is the default, and uses the default link weights that match the network topology. The other three types match the approaches of random, delta, and hybrid as discussed in previous chapter. We



*Figure 14 - PC vs. Approach, Abilene Topology*

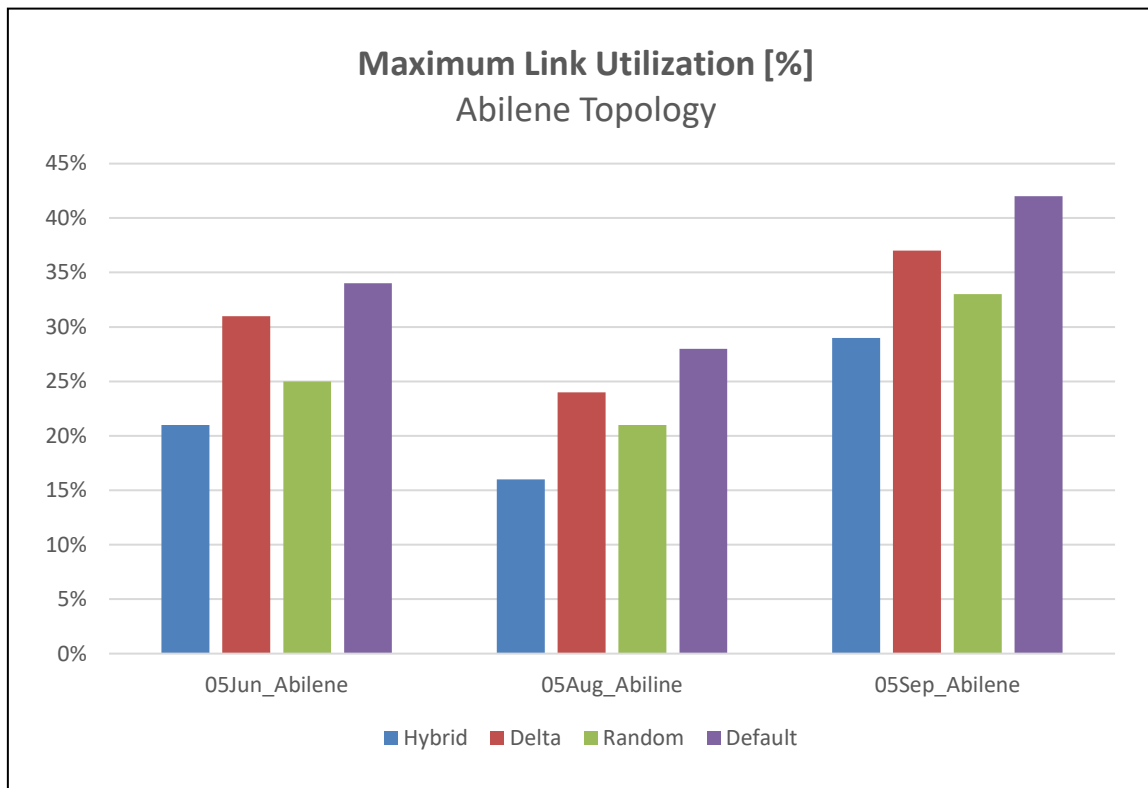
selected data sets of Abilene network topology from the months of June, August, and September of the year of 2004 (Tune et al., 2013).

Power consumption results from the three approaches is mainly affected by the assumption that a node can be put to sleep entirely if all of the incoming and outgoing flows through the node are negligible. When a node is turned off with its links, then its power consumption is zero. Neglecting the traffic is based on comparing the amount of



MLU to a lower limit (we selected 5% as lower limit). This has a tremendous effect on the total power of the network.

Across all traffic simulations, we keep track of the MLU, while turning off those nodes that have low link utilization. In order not to cause high MLU and not to cause congestion, as we mentioned in Chapter 3 we do turn on “top” node which possesses the highest incoming and outgoing traffic that can resolve the extra capacity required in the network (Francois et al., 2014).



*Figure 15 - MLU vs. Approach, Abilene Topology*

The performance of the three sets of traffic patterns using the LF TE scheme is demonstrated in Table 4. It shows the amount of savings in power consumption and MLU for each traffic pattern. All three approaches are compared to the results obtained when we use the default link weights. In the case of hybrid approach, the saving for the

September traffic pattern can reach 35.24% in power consumption savings while the MLU is also reduced by 30.95%.

Table 4 – *PC and MLU savings for three sets of traffic, Abilene topology*

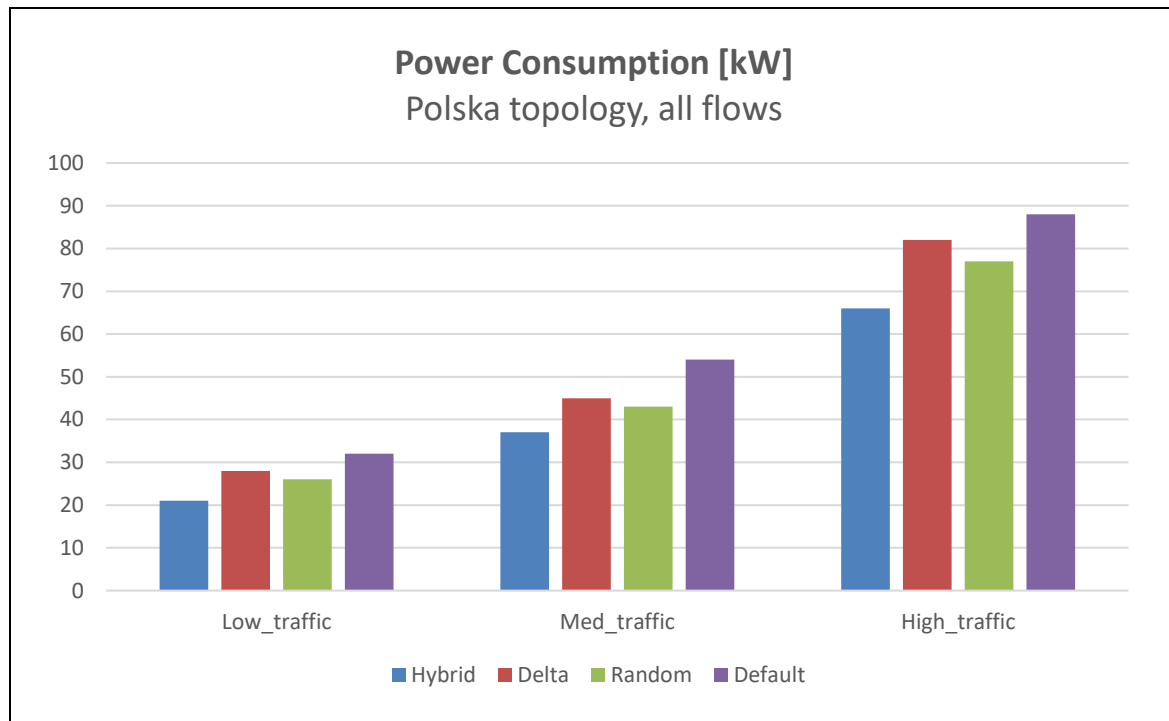
<i>Abilene</i> <i>Traffic period</i>	<b>PC Savings vs. Default</b>			<b>MLU Savings vs. Default</b>		
	Random	Delta	Hybrid	Random	Delta	Hybrid
<b>June</b>	25.68%	16.51%	33.09%	26.47%	8.82%	38.24%
<b>August</b>	20.00%	16.68%	32.45%	25.00%	14.29%	42.86%
<b>September</b>	29.75%	15.03%	35.24%	21.43%	11.90%	30.95%

We notice that delta weight approach has the lowest savings when compared as a standalone approach giving savings in the range of 8.82%-14.29% in MLU relative to the default settings. We get better solutions for the random approach since it has larger search space than the delta approach. Therefore, the first random approach gives much better savings in MLU and PC than the delta approach. Eventually combining both approaches of random and delta into the hybrid approach gives the best savings in MLU, and PC.

#### *Polska Network Topology Results*

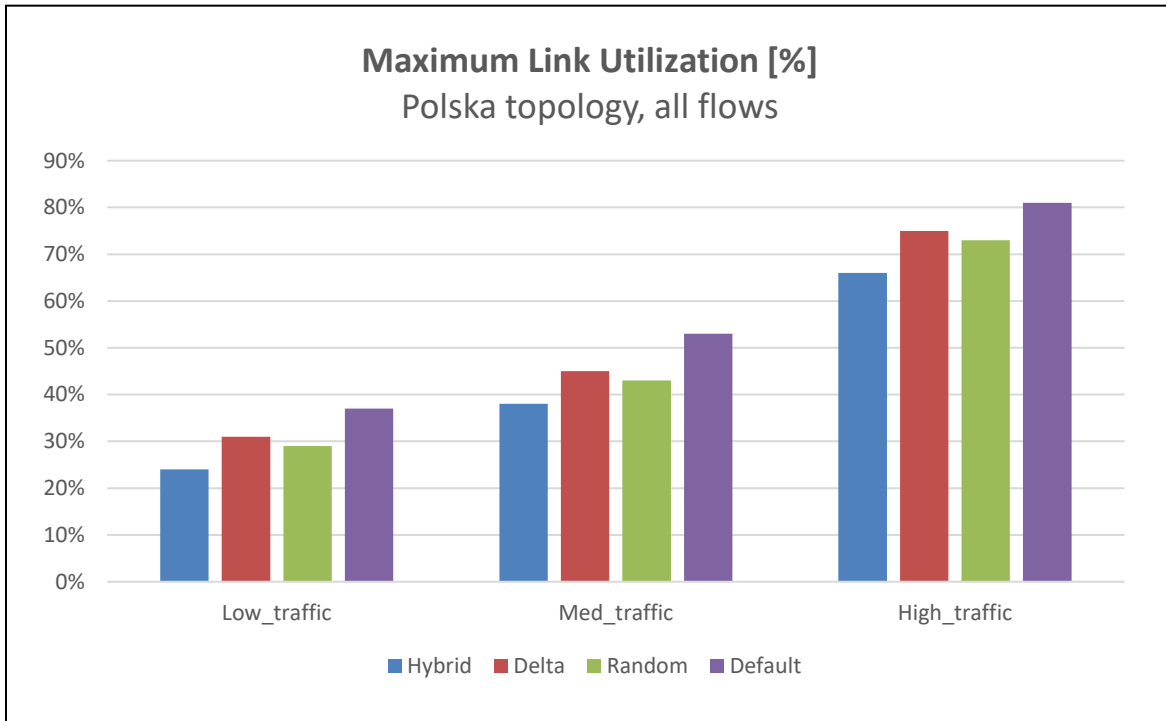
Our second network topology that we used for testing our research approach is network topology of Polska. In order to vary the synthetic meta-heuristics traffic for Polska network topology, we created three flavors of traffic which are deployed using LF TE scheme. The three traffic flow types vary the link load utilization by using low, medium or high traffic load. Simulation validation for those three traffic profiles includes various types of traffic patterns, such as Poisson, Periodic and Flat flows.

The default approach was chosen to have equal link weights for all links, and we selected a values of 32500, which is almost half the range allowed for any link weight as described in the work of (Fortz & Thorup, 2000). The three other approaches are the random, delta and hybrid approaches that we discussed in Chapter 3. In Figure 16, we show the simulations results of the power consumption (PC) for the four scenarios. We can see that the power consumption is much smaller in the case of the low traffic scenario. The hybrid approach gives the highest power savings, especially when we have high traffic scenario.



*Figure 16 - PC vs. Approach, Polska Topology*

Figure 17 describes the MLU obtained for the four different approaches that we simulated. The hybrid approach gives the lowest MLU value.



*Figure 17 - MLU vs. Approach, Polska Topology*

Table 5 summarizes the combined results from all simulations conducted. We used three traffic flows that mimic various link load scenarios from low to high network link loads. The range of link loads that we selected was based on altering the traffic flows using MGEN so that we keep the link load utilization between 30% and 90%. We measured link utilization such as low traffic scenario can reach up to 35%, while medium scenario matches 55% and high link load utilization should be around 85%. Using MGEN, we had chosen about 15, 25 and 30 traffic flows to be used for low, medium and high scenarios, respectively.

We can see from the Table 5 that as we increase the traffic level from low to high, the savings of MLU for the hybrid approach is decreased from 35.14% down to 18.52%. For the power savings, we do get similar savings from 34.38% down to 25%.

Table 5 - PC and MLU Savings for three sets of traffic, Polska Topology

<i>Polska</i> <i>Traffic Flow</i>	<b>PC Savings vs. Default</b>			<b>MLU Savings vs. Default</b>		
	Random	Delta	Hybrid	Random	Delta	Hybrid
<b>Low_traffic</b>	18.75%	12.50%	34.38%	21.62%	16.22%	35.14%
<b>Med_traffic</b>	20.37%	16.67%	31.48%	18.87%	15.09%	28.30%
<b>High_traffic</b>	12.50%	6.82%	25.00%	9.88%	7.41%	18.52%

Polska topology results show that the hybrid approach is more effective than the random and delta weight approaches.

### Network Flow Instability Measurements and Experiments

Extensive experiments for studying the effect of delay on network flow instability were carried out as shown below. As described in Chapter 3, while developing the approaches for reducing MLU and PC simultaneously as a multi-objective problem, we noticed that the traffic flow does not stabilize immediately after changing the link weights. This latter observation drove us to investigate what can be done in order to reduce this instability in the measurements of network flow after we apply the optimized link weights in every iteration.

The sections below describe the results of network instability. Based on equations 10 and 11 from Chapter 3, we performed measurements of the CORE network system in order to obtain  $average\_ \delta_{instability}$ . We run the same simulation several times and then take the average of the measurements. In the next sections, we show the results of our

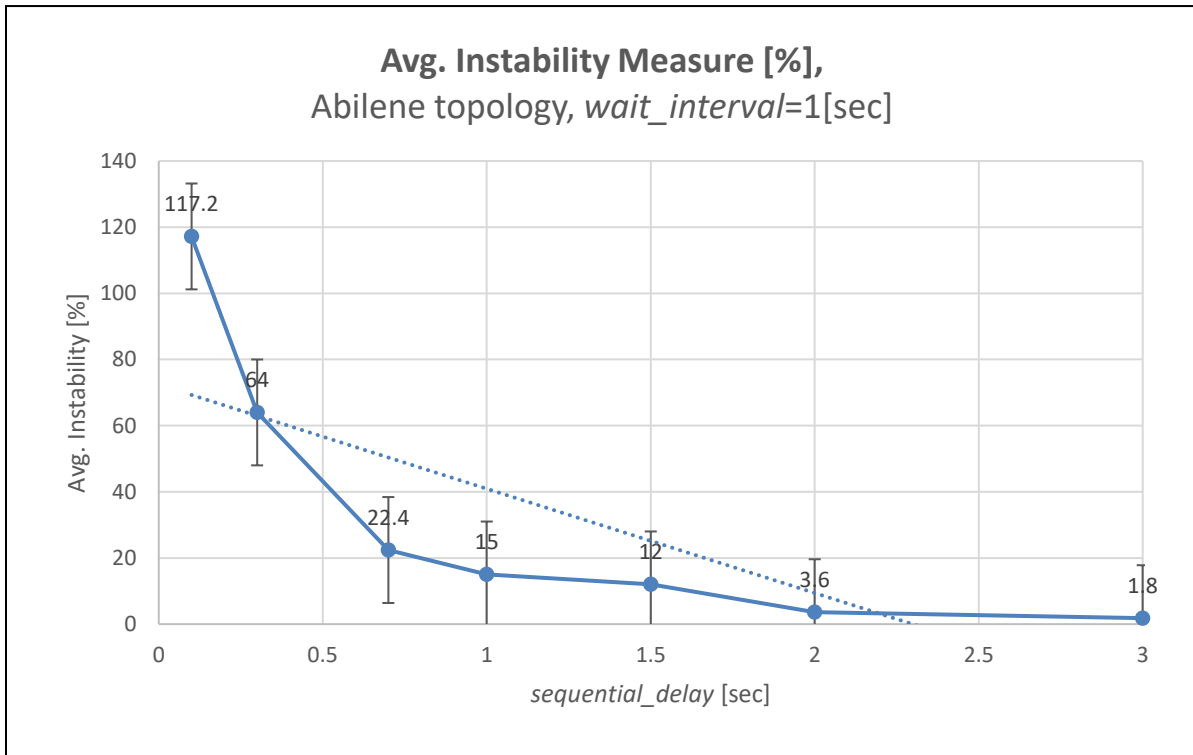
simulations while varying both types of delay parameters; *sequential\_delay* and *wait\_interval* delay.

The same two network topologies that were used for optimizing the MLU and PC were also used for verifying the instability study. The simulation used in this work adopted the best value of  $\Delta w$  from earlier simulations, and it used the same Python DEAP software parameters such as number of GA generations and mutation and crossover probabilities.

For the purpose of evaluation of this instability study, we used the hybrid approach for both network topologies.

#### *Sequential Delay Effect on Instability*

For testing the effect of *sequential\_delay* that is applied after each link weight alteration, we assumed that the *wait\_interval* delay is constant, and only varied the *sequential\_delay* from 0.1sec until 3sec as shown in Figure 18.



*Figure 18 - Avg. Instability Measure, Abilene Topology, *wait\_interval*=1sec*

We setup the same *wait\_interval* of 1sec, after applying all link weights for every node, and the *sequential\_delay* was varied from 0.1sec until 3sec. Figure 18 shows how the average measured instability is reduced as we increase the *sequential\_delay*. The average instability measure shows that we converge to a 22.4% already when we increase the *sequential\_delay* from 0.1sec until 0.7sec. After 0.7sec we see that the instability measure improves but it already reached closer to lowest value of 1.8%.

Figure 19 shows what happens if we increase the *wait\_interval* from 1sec to 5sec. This allows the CORE network to reduce oscillations even much faster with respect to increasing the *sequential\_delay* in every iteration. As we see in Figure 19 we already reached 18.2% of instability when the *sequential\_delay* decreased down to 0.7sec. In the end of simulation when the *sequential\_delay* is increased to 3sec we can reach 1.4%.

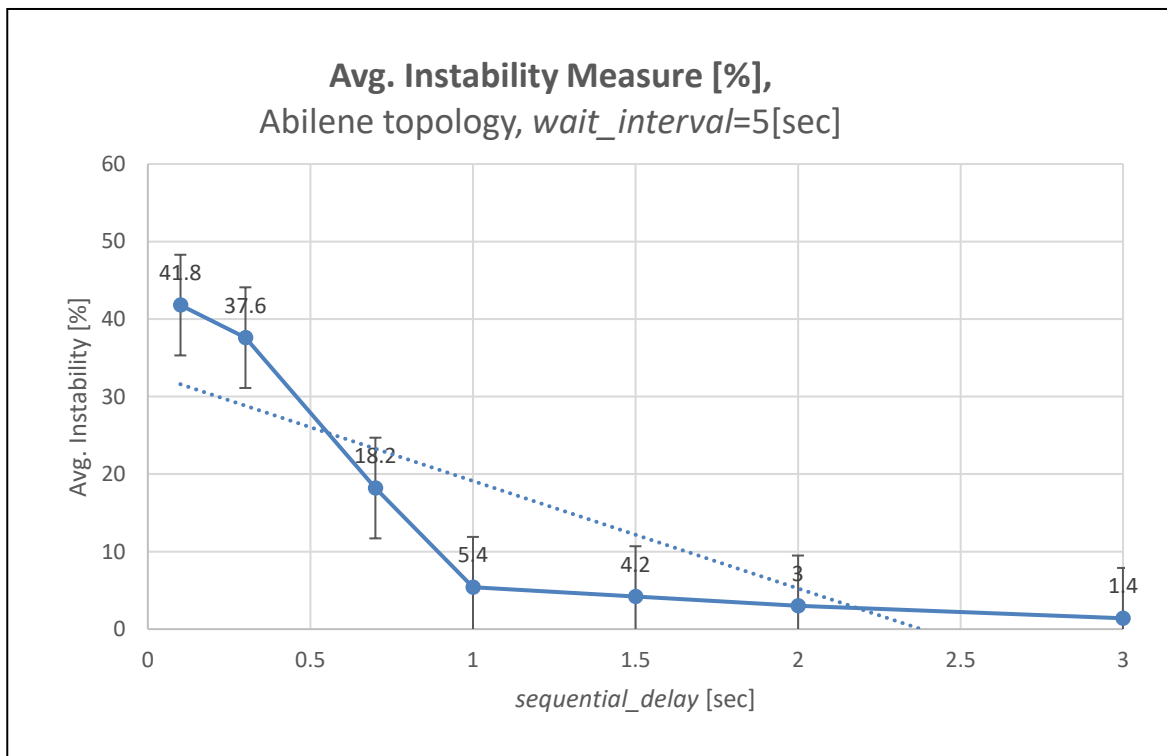


Figure 19 - Avg. Instability Measure, Abilene Topology, *wait\_interval*=5sec

The dotted line shown in Figure 18 and Figure 19 represents the trend line that is calculated by the spreadsheet tool based on the values of the data points.

We performed similar experiments on Polska topology with medium traffic flow configured by MGEN. Both *wait\_interval* and *sequential\_delay* parameters were varied similarly as we presented for Abilene network topology.

Figure 20 shows the average instability measurement in percentage as a function of *sequential\_delay* varied from 0.1sec until 3sec, when the *wait\_interval* is set to 1sec.

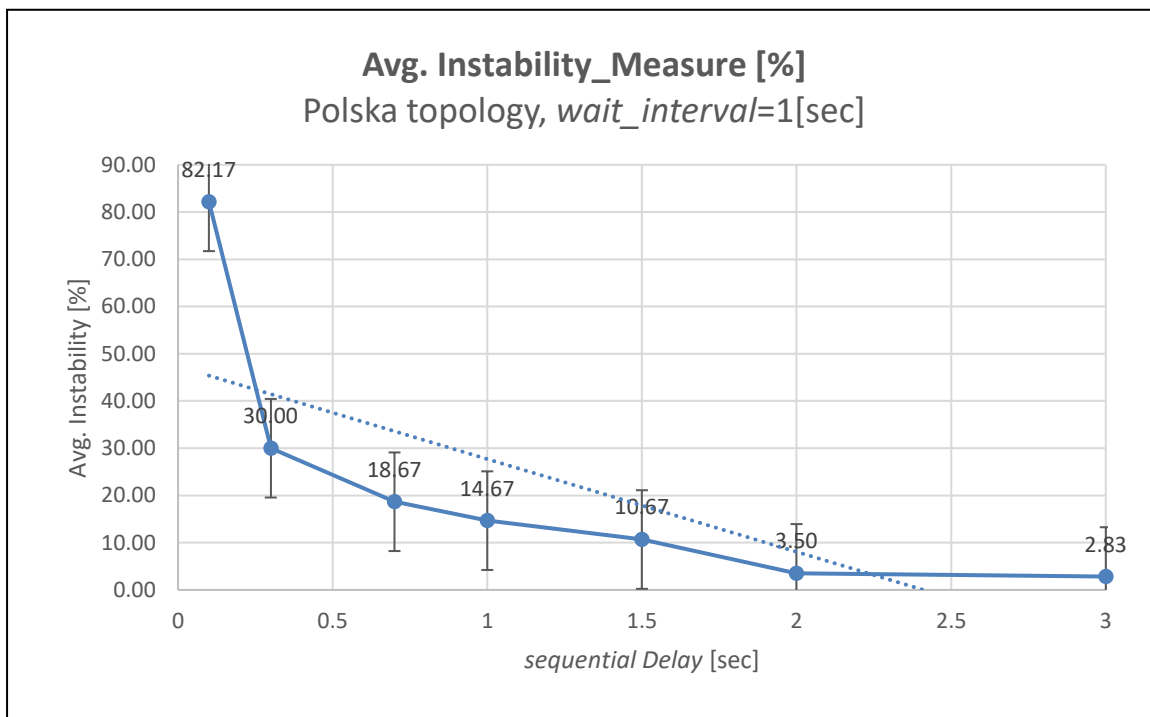


Figure 20 - Avg. Instability Measure, Polska Topology, *wait\_interval*=1sec

In Figure 21, we show how the instability measurement of Polska topology changes as we increase the *sequential\_delay* when *wait\_interval* is set to relatively large value such as 5sec.



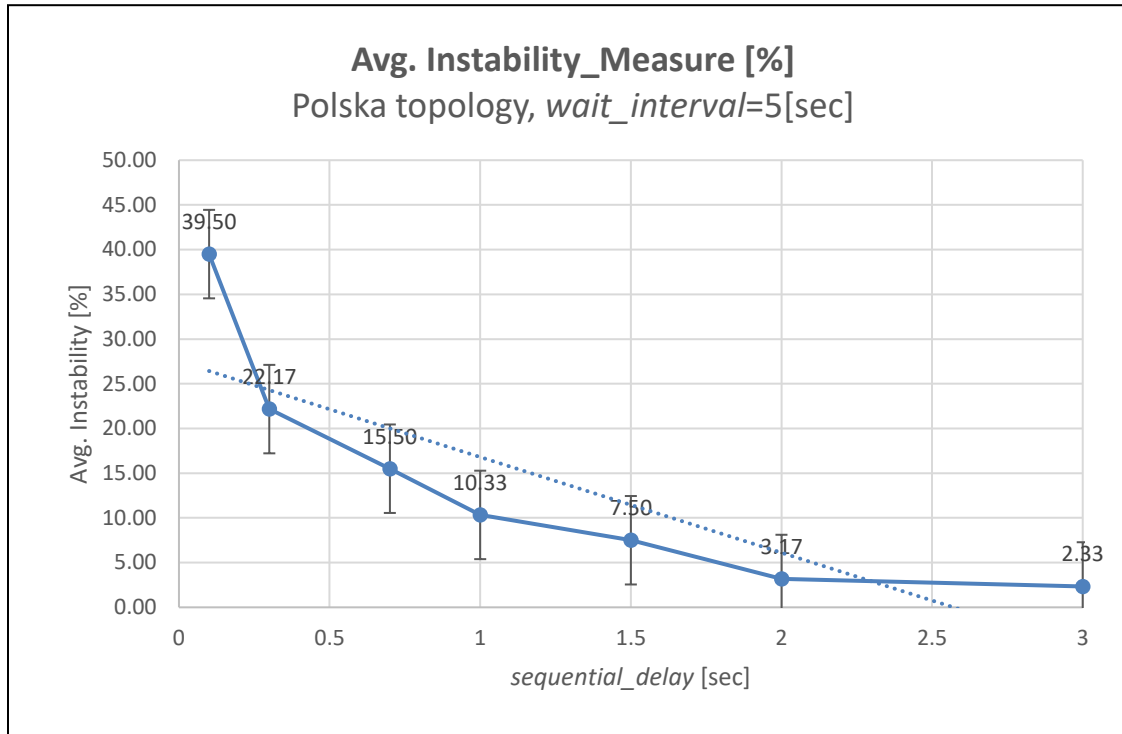


Figure 21 - Avg. Instability Measure, Polska Topology, *wait\_interval*=5sec

#### *Wait Interval Delay Effect on Instability*

We performed similar experiments on both network topologies of Abilene and Polska, where we varied the *wait\_interval* delay while the *sequential\_delay* was kept constant.

*Wait\_interval* delay was increased from 0.5sec until 5sec.

Figure 22 shows the results of varying *wait\_interval* delay using Abilene topology and traffic from August of 2004. As we can see that the larger the *wait\_interval*, the lower instability we obtain. The slope of the trend line is not steep enough, and shows that the effect of increasing the *wait\_interval* delay is not as effective as we noted in Figure 20

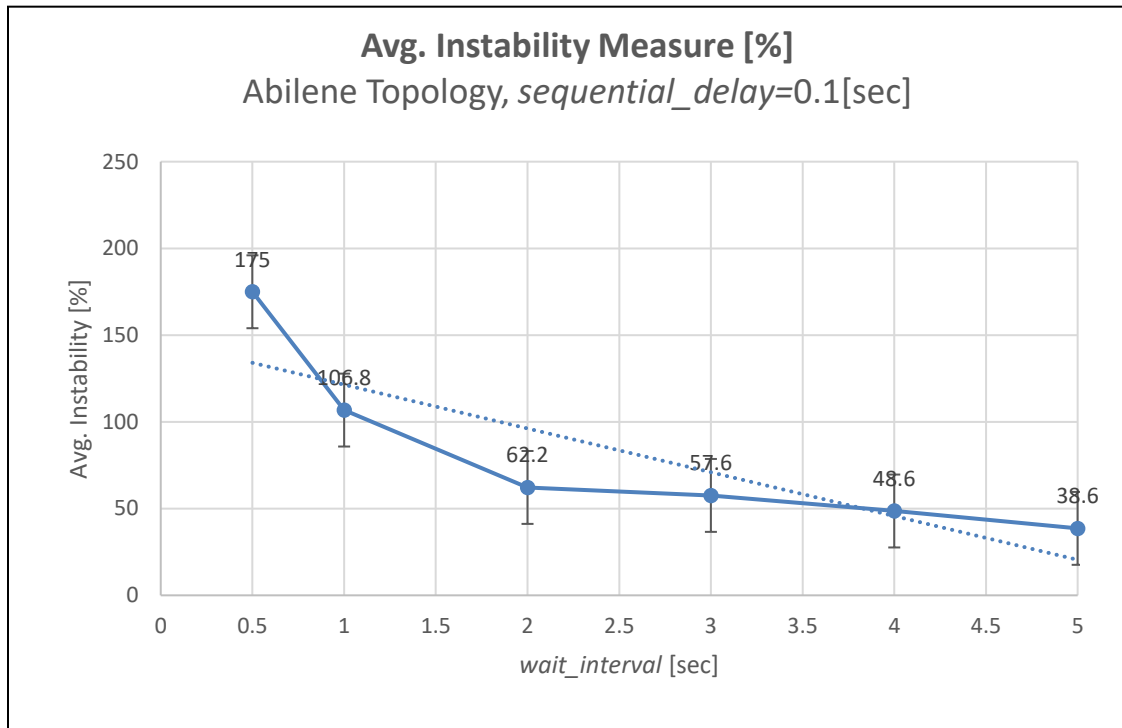


Figure 22 - Avg. Instability Measure, Abilene Topology, *sequential\_delay*=0.1sec

where the *sequential\_delay* increase up to 0.7sec is more effective in reducing the instability down to 18.67%.

Increasing the *sequential\_delay* to 1sec shows that instability is drastically improved even when the *wait\_interval* is 0.5sec. Figure 23 shows the effect of increasing *wait\_interval* delay from 0.5sec to 5sec on the instability measure.

As we note in earlier results, the optimum delay used for *sequential\_delay* is 0.7sec while the *wait\_interval* delay best value was chosen to be 2sec.

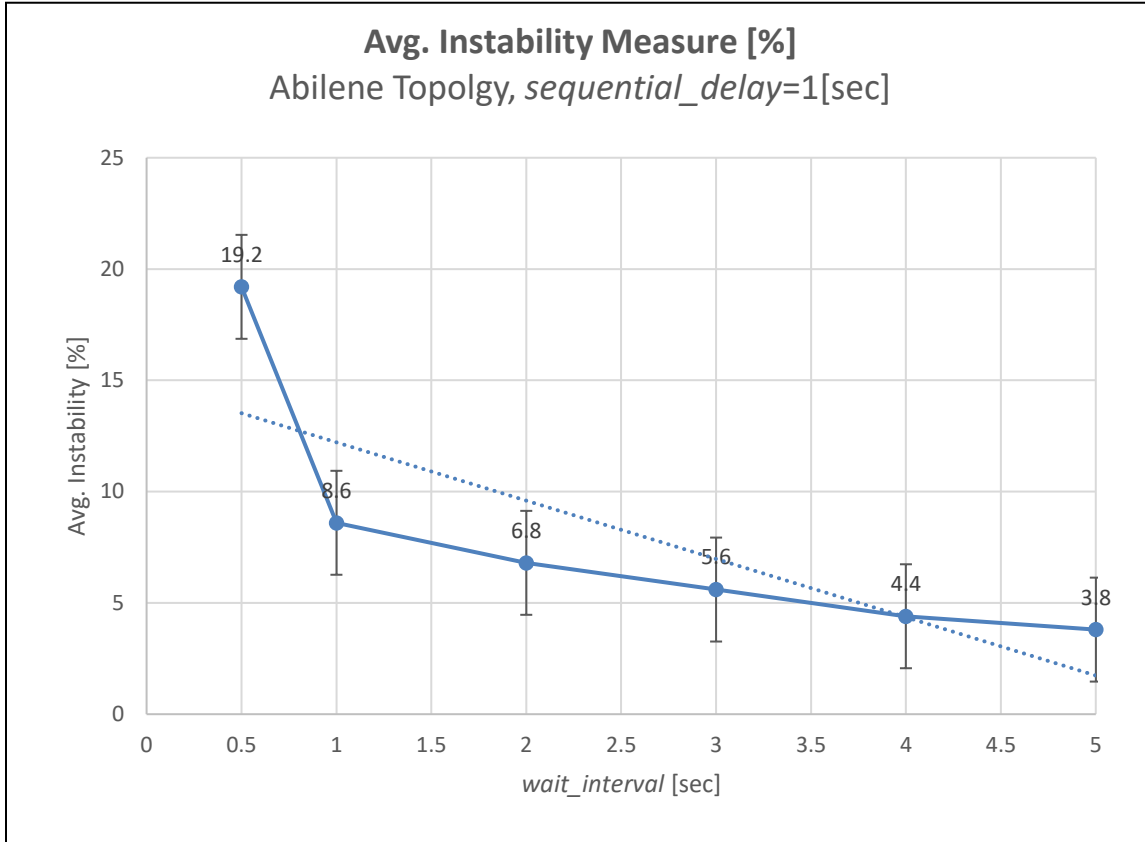


Figure 23 - Avg. Instability Measure, Abilene Topology, *sequential\_delay*=1sec

In the case of Polska topology, we performed similar experiments while using medium traffic flow generated by MGEN. In Figure 24, we show the average instability measure with constant *sequential\_delay* set at 0.1sec, used to apply each link weight for every node in the network, while varying the *wait\_interval* from 0.5sec until 5sec.

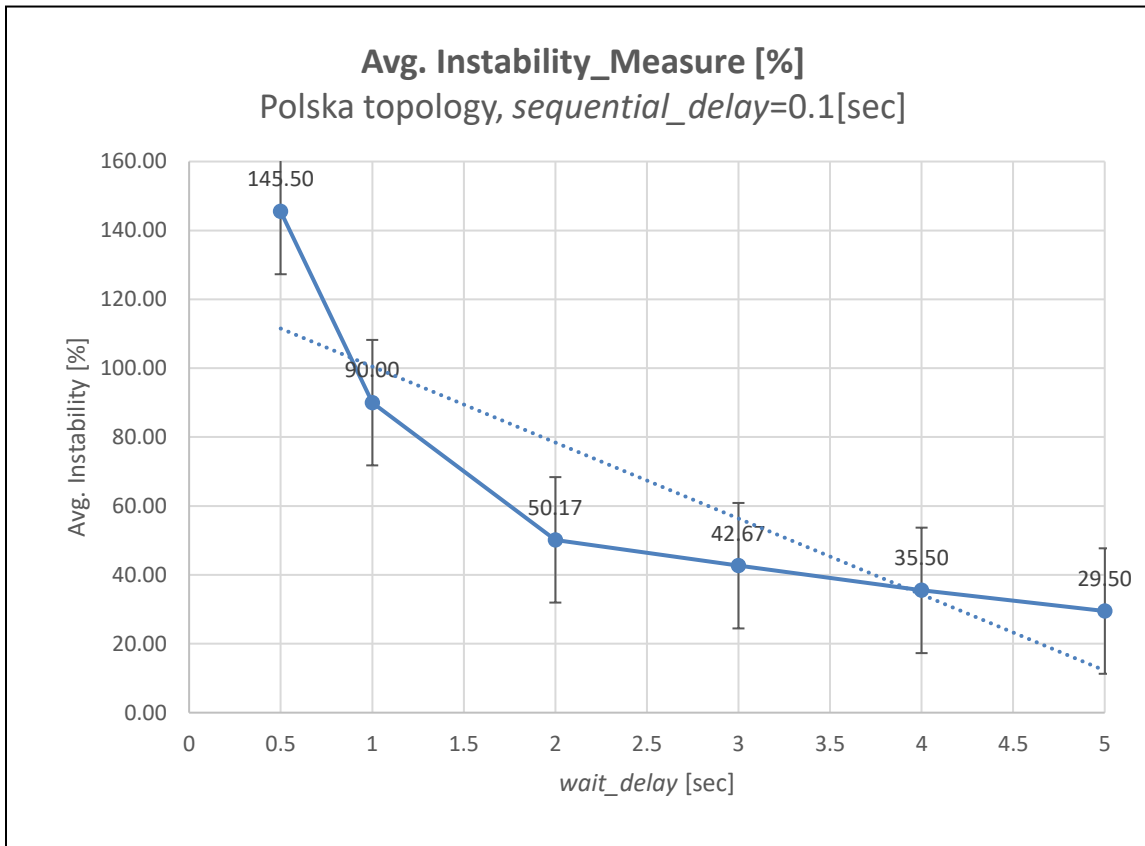


Figure 24 - Avg. Instability Measure, Polska Topology, *sequential\_delay*=0.1sec

Figure 25 shows a similar graph as in Figure 24, with fixed *sequential\_delay* of 1sec, while varying the *wait\_interval* delay from 0.5sec until 5sec. From both Figure 24 and Figure 25, we notice that increasing the *sequential\_delay* from 0.1sec to 1sec has a tremendous effect on the instability measure. We already achieve a measure of 17% when the *wait\_interval* increases to 2sec, while in the case of 0.1sec *sequential\_delay* we have 3 times worst instability measurement.

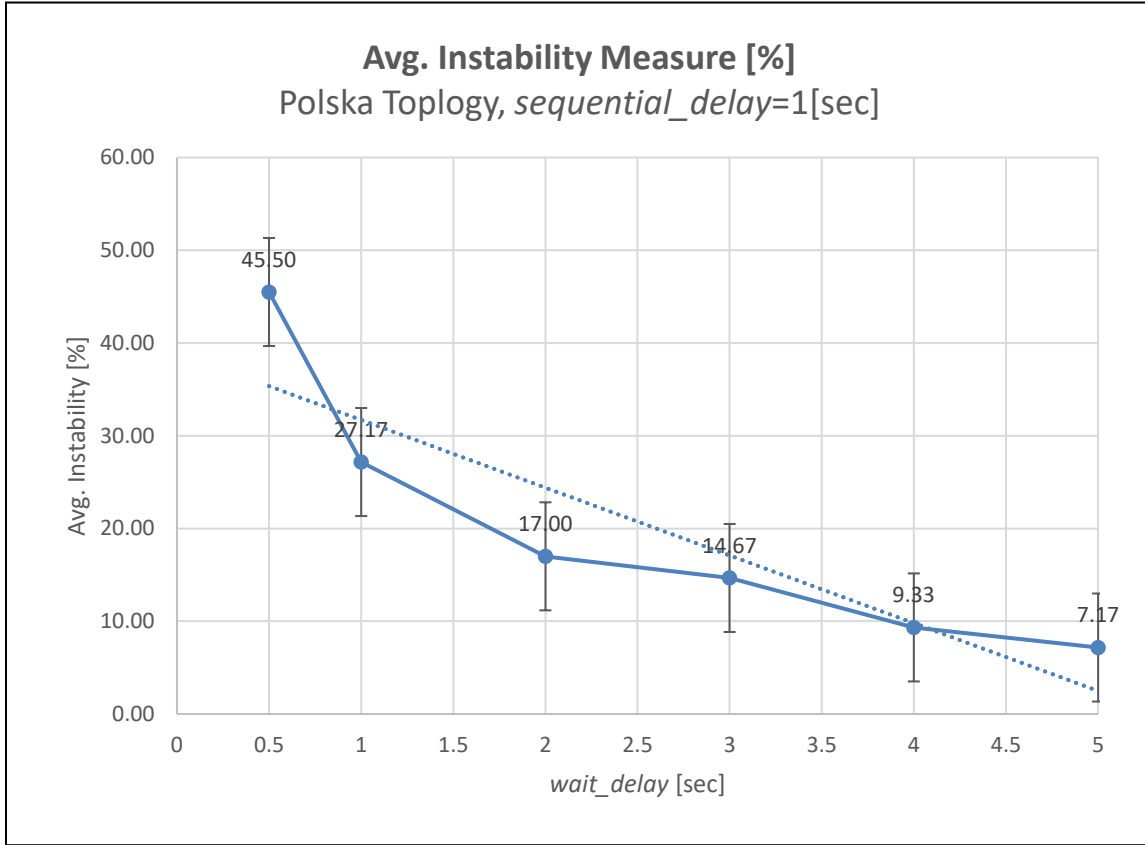


Figure 25 - Avg. Instability Measure, Polska Topology, *sequential\_delay*=1sec

Overall, both Polska and Abilene topologies exhibit similar results when it comes to *sequential\_delay* usage, and stability of network flow is achieved when increasing *wait\_interval* delay to 2sec. As mentioned before, due to random nature of the GA algorithm, results vary from one experiment to the next, but we ran the experiments at least 10 times, and then we averaged the results in order to present them in this research.

#### **$\Delta w$ vs. Instability Measurement**

Earlier in Chapter 4, we discussed how the  $\Delta w$  was chosen in order to minimize both objectives of MLU and PC in the case of using the  $\Delta w$  and the hybrid approach. For testing the instability of network with respect of  $\Delta w$ , we carried several experiments and measurements in order to check how this network instability is affected by changing  $\Delta w$ .

Figure 26 shows how Abilene network topology MLU and PC vary as a function of  $\Delta w$  values. It also shows how the instability measure is changing as  $\Delta w$  increases. For these simulations, we used a 1sec delay value for both the *sequential\_delay* and the

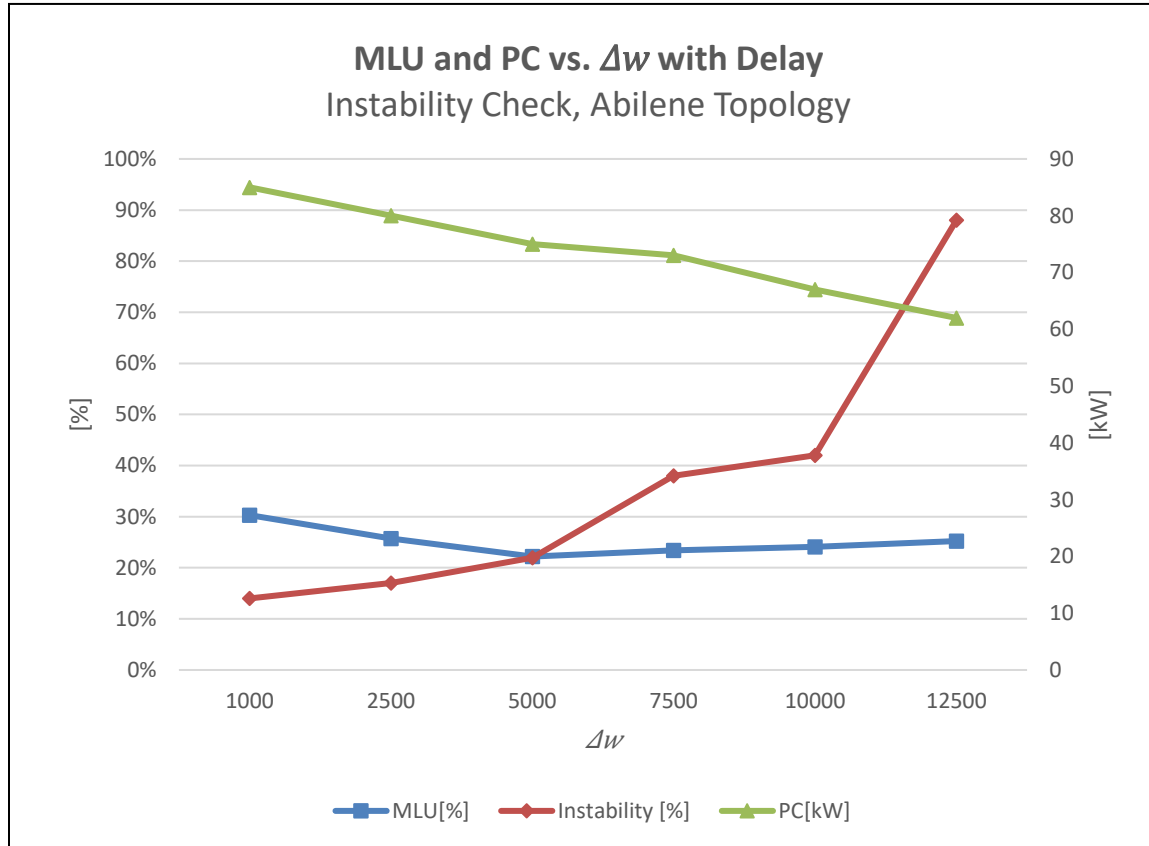


Figure 26 - MLU and PC vs.  $\Delta w$ , with Delay and Instability, Abilene Topology

*wait\_interval* delay parameters. From instability measurements, it makes sense that the lowest value is obtained when  $\Delta w$  is smallest, but as  $\Delta w$  increases, we notice that the instability increases until it reaches 88% when  $\Delta w$  reaches 12500. Best MLU and PC values are obtained when  $\Delta w$  is around 5000, and we notice that MLU and PC decreased by at least 5% for MLU and 7 [kW] for PC compared to the hybrid approach results. The best  $\Delta w$  of 5000 is in line with our observation in Figure 12 earlier in this chapter. When we added delay parameters then instability improved, and thus the MLU and PC

measurements yielded lower numbers. In Figure 12, we did not use additional wait and sequential delays in our simulations, which caused instability to increase and thus yielded worst MLU and PC. On the other hand, the MLU and PC trend line versus  $\Delta w$  value is similar to what we found in Figure 12.

In the case of Polska topology with medium traffic flow, Figure 27 shows the results of varying  $\Delta w$  on MLU, PC, and instability measure. We used the same delay value of 1sec as in Abilene network topology in Figure 26 before, for both the *sequential\_delay* and the *wait\_interval* delay parameters. This result emphasizes further the results obtained in Abilene topology, where the instability measure is increased as  $\Delta w$  is increased, and the optimized values of minimum MLU and PC are achieved when  $\Delta w$  is around 5000 similar to what we obtained in the case of Abilene topology. Increasing the  $\Delta w$  to 12500 causes average instability measurement to increase beyond 75%. Similarly

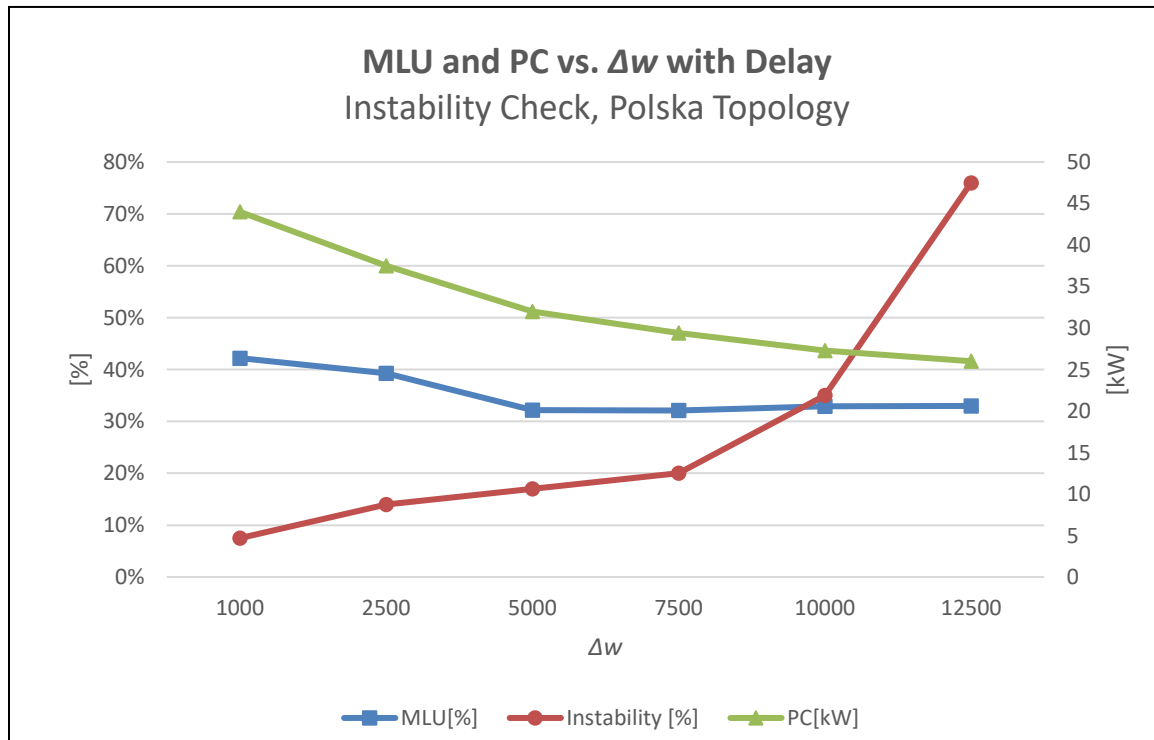


Figure 27 - MLU and PC vs.  $\Delta w$  with Delay and Instability, Polska Topology

as observed in Figure 26 and Figure 12, we also see similar observation between Figure 27 and Figure 13 for Polska topology, where MLU and PC become worst when we do not use wait and sequential delays for reducing instability.

Figure 28 and Figure 29 show the instability simulations results, when zero delay is

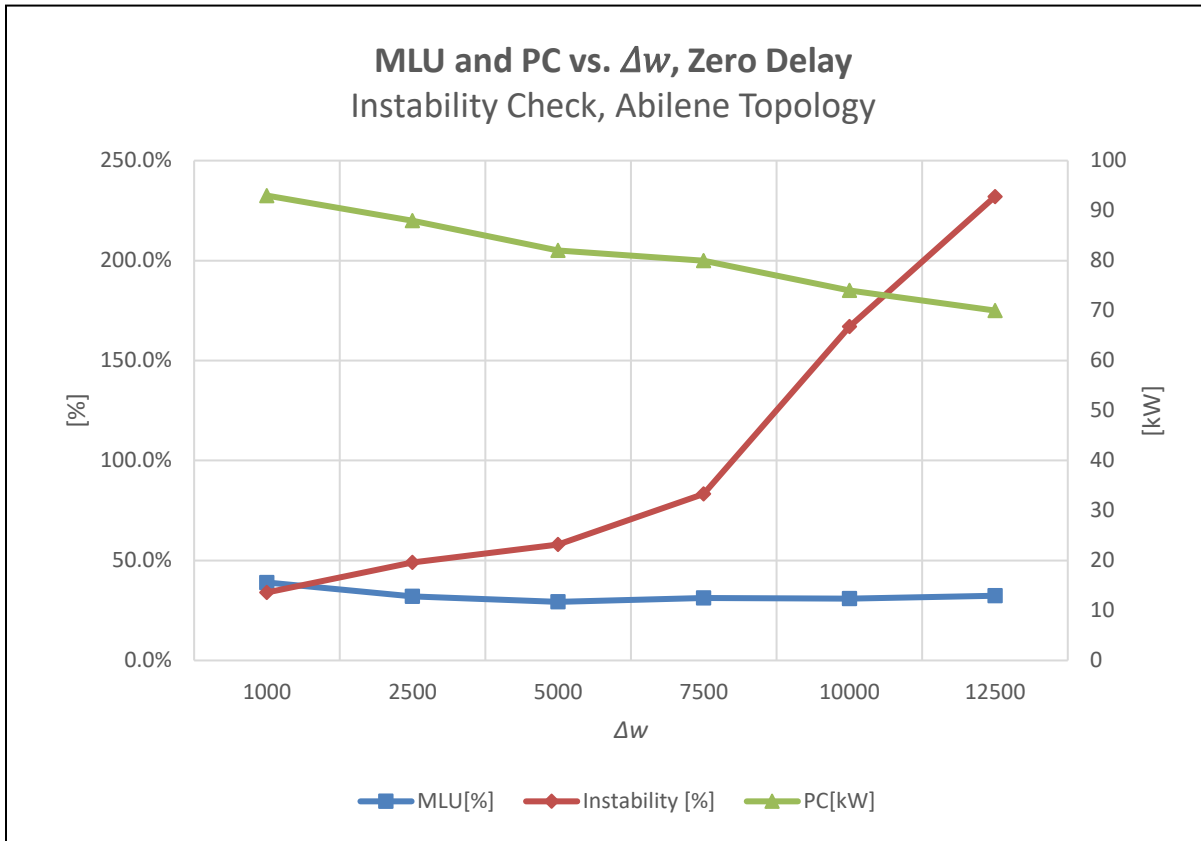


Figure 28 – Instability, MLU and PC vs.  $\Delta w$  with Zero Delay, Abilene Topology

used for *sequential\_delay* and *wait\_interval* parameters, respectively for Abilene and Polska topologies. The MLU and PC results are similar to Figure 12 and Figure 13 that we showed for Abilene and Polska topologies, respectively.



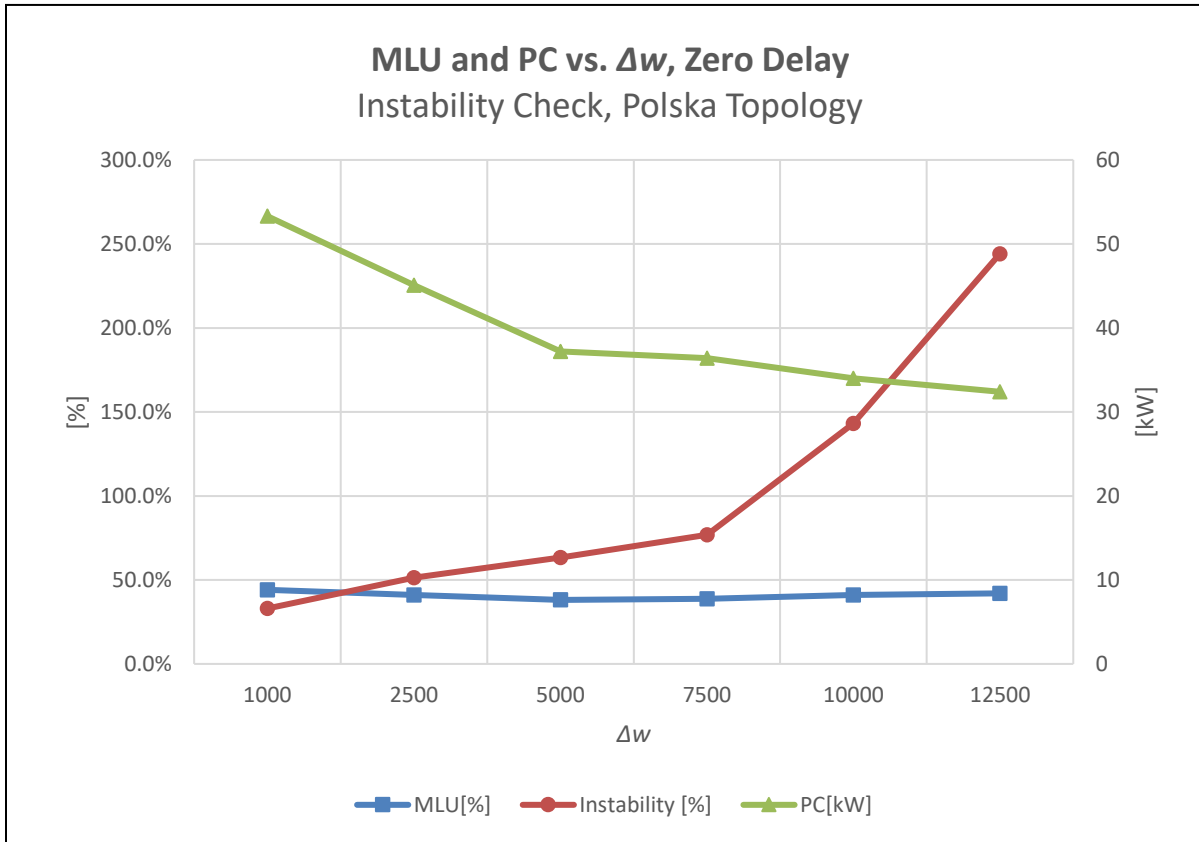


Figure 29 - Instability, MLU and PC vs.  $\Delta w$  with Zero Delay, Polska Topology

We noticed from Figure 28 and Figure 29, that instability measure is much worse than when we use delay parameters in our simulations. In addition, the MLU and PC obtained results are worse in the case of zero delay than when we use 1sec delay parameters, as mentioned in our earlier discussion. In the next sub section, we examine the MLU and PC values obtained when we use 1sec for the delay parameters of *sequential\_delay* and *wait\_interval*.

Overall, the outcome of instability simulations show consistent results when we increase  $\Delta w$  across the two network topologies and using traffic patterns that we chose. We also managed to improve instability when the delay parameters were increased.

Furthermore, the trend of instability results continued to be similar in both cases when using zero delay for *wait\_interval* and *sequential\_delay* and when using 1sec for both delay parameters.

#### *Final Results Including $\Delta w$ and Best Instability Measurement*

As we indicated before, increasing the  $\Delta w$  parameter yielded better results and we obtained good solutions for PC and MLU. The downside is that network oscillations increase and average instability rises. Applying *wait\_interval* and *sequential\_delay* reduced the average instability, and thus it allowed us to retrieve better and more reliable PC and MLU measurements with stable traffic. Using the same optimum  $\Delta w$  value of 5000 gave better solutions that produced lower values of MLU and PC than what we obtained when we did not implement *wait\_interval* and *sequential\_delay*.

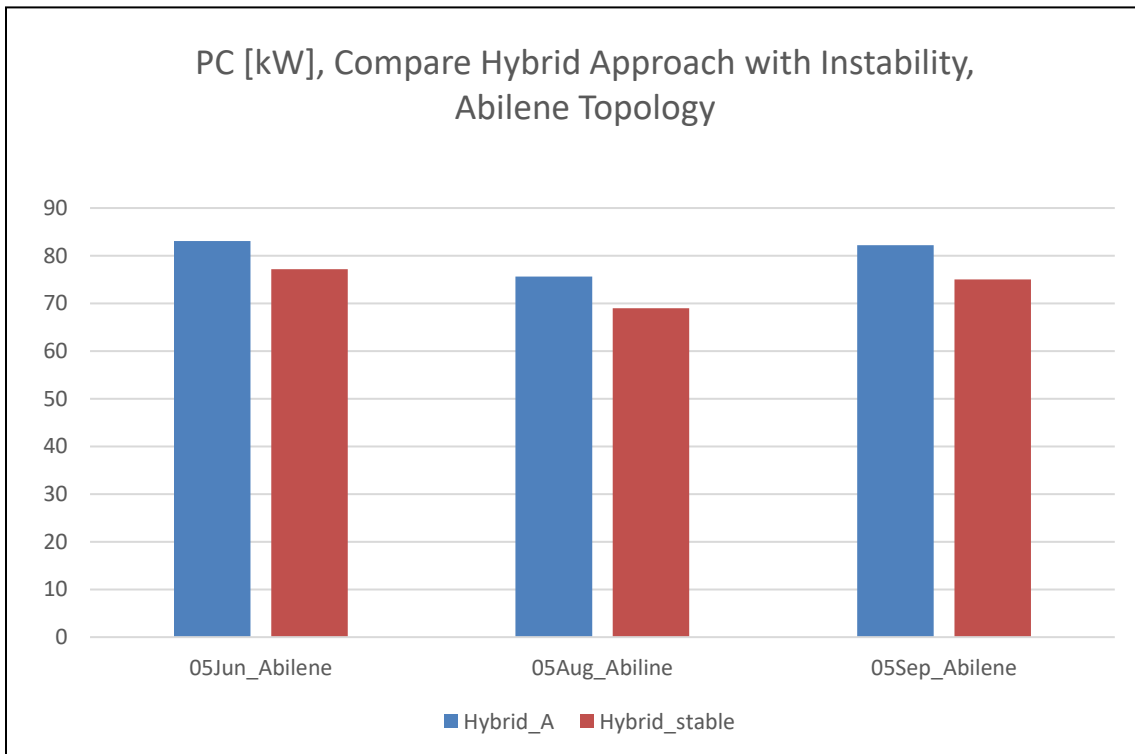


Figure 30 - PC Comparison of Hybrid Approaches with Instability, Abilene Topology

In Figure 30 and Figure 31 we show PC and MLU results of Abilene network simulations, respectively after we increased the delay parameters to 1sec. We compare the hybrid approach that we presented earlier with a new approach called *hybrid\_stable* that uses *wait\_interval* and *sequential\_delay* in order to reduce the average instability and oscillations in the network. In fact, the hybrid approach is the same as the *hybrid\_stable* approach with *wait\_interval* and *sequential\_delay* parameters set to a value of 0. The MLU improved further by 20.7% in the case of 05\_September Abilene topology traffic case, while the PC was reduced further by 8.8% beyond what the hybrid approach gave us earlier.

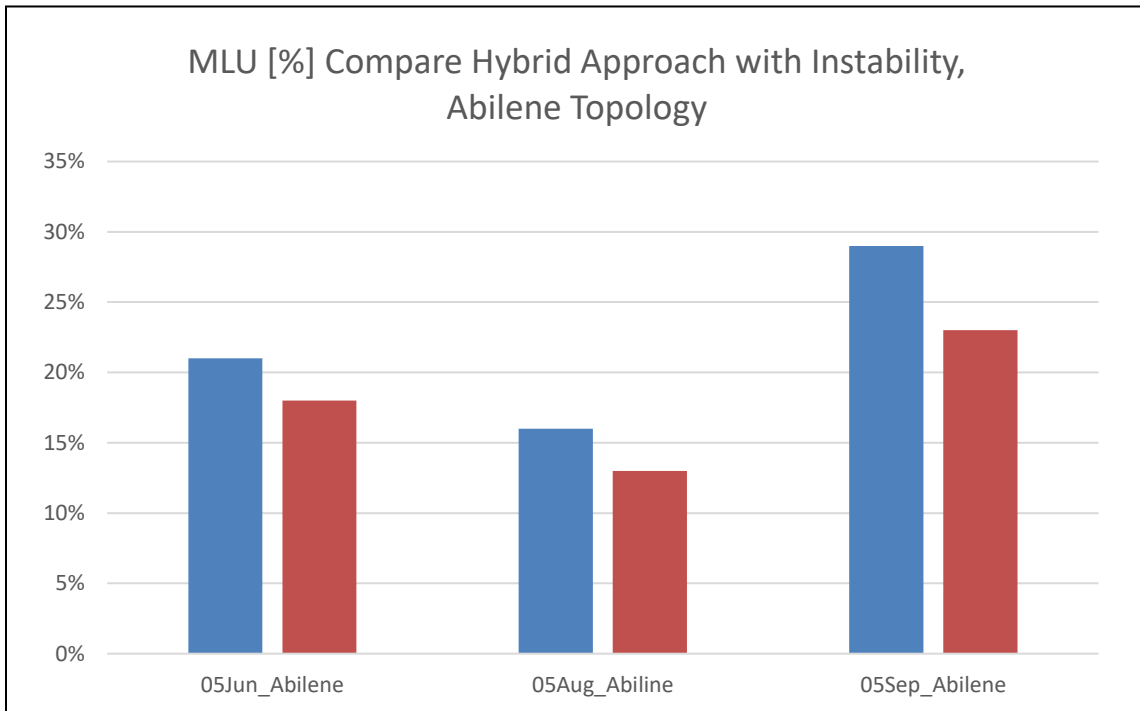


Figure 31 - MLU Comparison of Hybrid Approaches with Instability, Abilene Topology

These simulations used the best *sequential\_delay* and *wait\_interval* of 1sec for both parameters, and average instability is at acceptable level of  $< 20\%$ .

In Figure 32 and Figure 33, we show the results of PC and MLU for Polska network topology, respectively.

Using Polska topology with the three different network traffic loads of low medium and high levels, we obtained better improvement in MLU and PC as the traffic load increases from low to medium to high. The hybrid approach was inferior to the hybrid\_stable approach when we used the delay parameters of *wait\_interval* and *sequential\_delay*. The best  $\Delta w$  is still 5000, similar to the experiments we did without applying delay parameters of *sequential\_delay* and *wait\_interval*. Maximum average

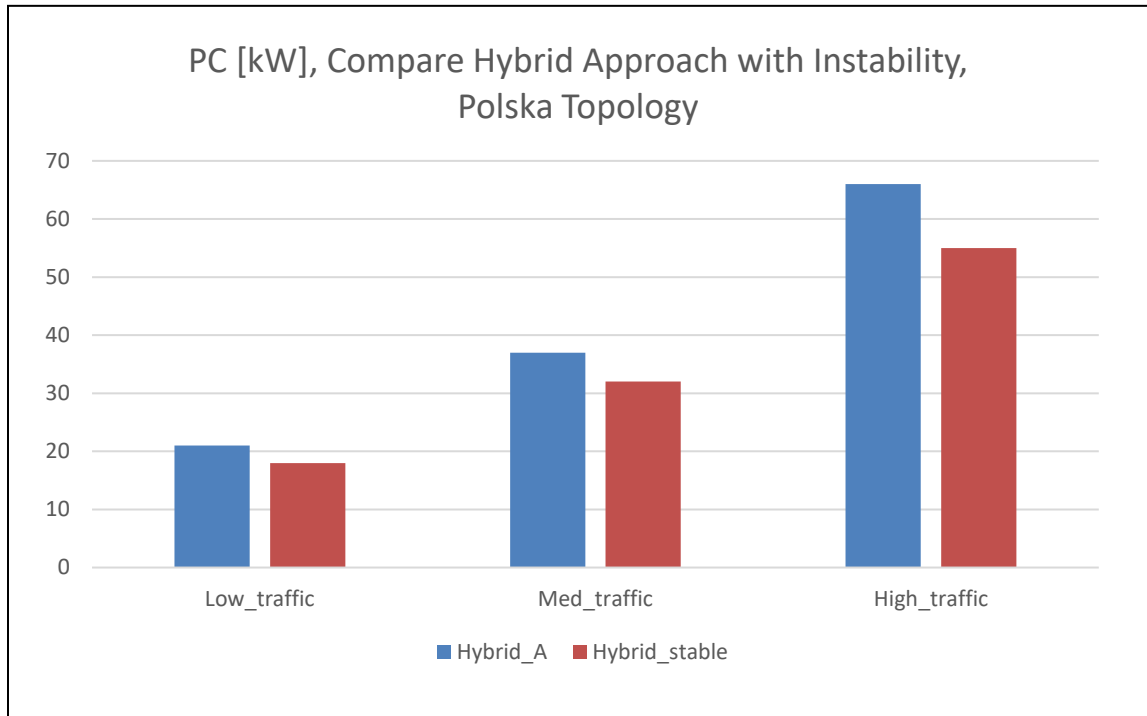


Figure 32 - PC Comparison of Hybrid Approaches with Instability, Polska Topology

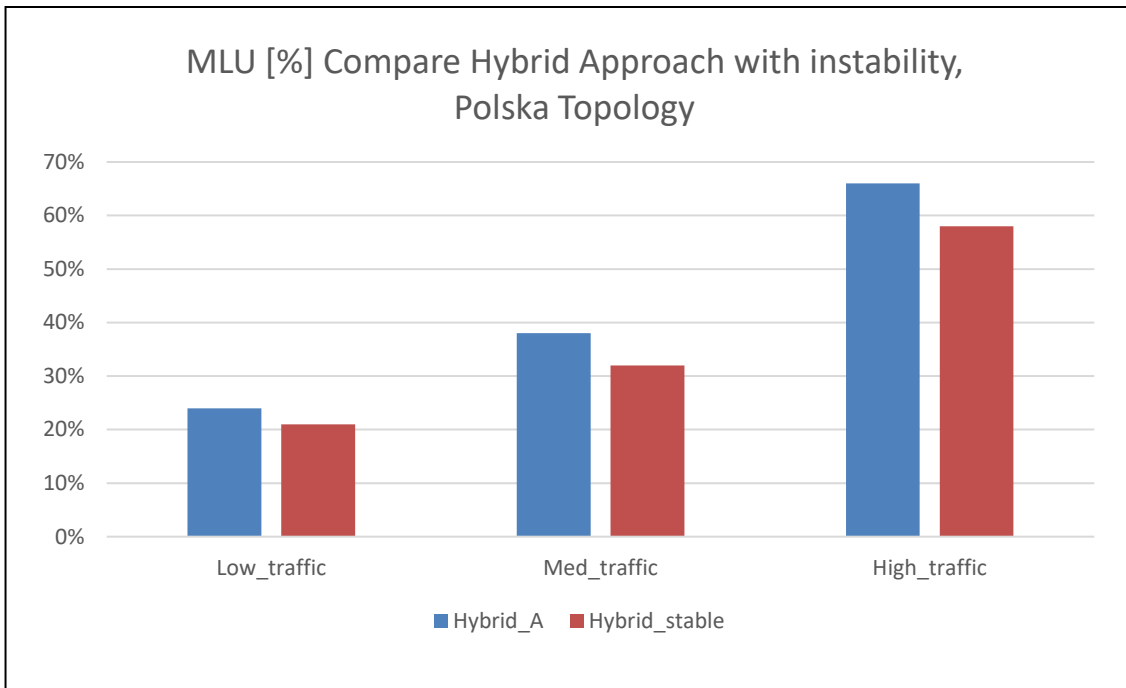


Figure 33 - MLU Comparison of Hybrid Approaches with Instability, Polska Topology

savings that we collected from many experiments reveal that we gain about 12% less MLU in the case of high traffic pattern, while the PC is reduced by 17%.

In Table 6 we summarize all the results for Abilene network topology, when we combine all 4 approaches discussed earlier compared to the default settings of the network.

Table 6 - Final Abilene PC and MLU savings with Hybrid\_stable approach

Abilene Traffic period	PC Savings vs. Default				MLU Savings vs. Default			
	Rando m	Delta	Hybrid	Hybrid _stable	Random	Delta	Hybrid	Hybrid _stable
<b>June</b>	25.68%	16.51%	33.09%	37.84%	26.47%	8.82%	38.24%	47.06%
<b>August</b>	20.00%	16.68%	32.45%	42.74%	25.00%	14.29%	42.86%	53.57%
<b>September</b>	29.75%	15.03%	35.24%	42.79%	21.43%	11.90%	30.95%	45.24%

For Polska network topology similar results are shown in Table 7, and we see that those results are consistent with the simulations with various *wait\_interval* and *sequential\_delay*. Increasing the  $\Delta w$  up to 12500 further continue to show similar trend as the other topology of Abilene, where instability and MLU get worst while the PC is reduced further.

Table 7 - Final Polska PC and MLU savings with Hybrid\_stable approach

<i>Polska</i> <i>Traffic Flow</i>	PC Savings vs. Default				MLU Savings vs. Default			
	Random	Delta	Hybrid	Hybrid_ stable	Random	Delta	Hybrid	Hybrid_ stable
<b>Low_traffic</b>	18.75%	12.50%	34.38%	43.75%	21.62%	16.22%	35.14%	43.24%
<b>Med_traffic</b>	20.37%	16.67%	31.48%	40.74%	18.87%	15.09%	28.30%	39.62%
<b>High_traffic</b>	12.50%	6.82%	25.00%	37.50%	9.88%	7.41%	18.52%	28.40%

### Summary of Results

All three approaches of random,  $\Delta w$  and hybrid, for optimizing the MLU and PC were presented in this chapter. We obtained consistent results showing that the best approach for reducing MLU and PC is the hybrid approach, when using two types of network topologies of Abilene and Polska. In the case of Abilene topology, MLU savings were in the range of 30.95% to 42.86% and PC was reduced from 33.09% to 35.24% for the hybrid approach. In the case of Polska topology MLU savings were in the range of 18.52% to 35.14% and PC was reduced 25% to 34.38% for the hybrid approach.

Instability measurements of the network show that we should apply both *sequential\_delay* and *wait\_interval* delays in order to achieve better network stability and reduce network oscillations. When using the best values of 1sec for the *wait\_interval* and 1sec for the *sequential\_delay*, we managed to get better instability and reduced further the values of MLU and PC for both topologies of Abilene and Polska.

A new approach called *hybrid\_stable* was introduced which combines the hybrid approach with the best values of *sequential\_delay* and *wait\_interval* numbers of 1sec. In the case of Abilene topology, the MLU improved further by 20.7% in the case of 05\_September traffic case, while the PC was reduced further by 8.8% beyond what the hybrid approach gave us when no delay parameters were used. In the case of Polska topology, we show that we gain about 12% less MLU in the case of high traffic pattern, while the PC is reduced by 17%.

Consequently, simulation runtime was affected when we used delay parameters in order to improve the stability of the network. Using *sequential\_delay* for every node iteration, and using constant *wait\_interval* delay after each GA iteration caused simulation time to increase.

## Chapter 5 - Conclusions

### Overview

We have presented three approaches for reducing power while maintaining acceptable MLU, in wired networks that use OSPF by performing link weight optimization. The first approach uses a random method for selecting the initial set of link weights, while the second approach uses  $\Delta w$  for varying the link weights, and in the third approach, a hybrid method is used by combining both the first and second approaches. All three approaches use GA MOO for finding new solutions that are applied to the network. GA solutions are obtained by using DEAP Python software and in each GA iteration, the best solution is tracked. Two network topologies of Abilene and Polska were used for testing the three approaches. Network traffic dataset for Abilene was selected from three different dates based on (Orlowski et al., 2010), while for Polska topology we created three different synthetic traffic loads. Best results were achieved by the hybrid approach and show that for Abilene network topology, we can reduce power consumption from 32.45% to 35.24%, while reducing MLU in average from 30.95% to 42.86% for specific traffic pattern. For Polska network topology, our hybrid approach showed a reduction in average from 25% to 34.38% in power consumption and MLU savings in the range of 18.52% to 35.14% for the medium traffic pattern. We also performed extensive study for measuring the effect of oscillations or network instability on the traffic flow measurements and reducing MLU and PC further by adding network delay parameters of *sequential\_delay* and *wait\_interval*.



This chapter draws the dissertation conclusions and implications towards the current standing of concurrent saving of Internet energy, and reducing MLU. It then discusses recommendations for future work and includes a summary of the chapter.

## Conclusions

The focus of this study was to answer our main research question, which is reducing network power consumption by applying our hybrid approach. The hybrid approach uses the best set of link weights obtained by the first approach as the initial solution, and then it uses the delta weight approach for finding better solutions. Our hybrid approach combines the advantage of using  $\Delta w$  for tuning the link weights, in conjunction with the random approach for selecting the initial link weights. We conducted several experiments to show that the hybrid approach could reduce power consumption (PC) further while reducing the MLU even more.

We answered the research question whether traffic changes can cause oscillations by conducting many experiments, where we altered the *sequential\_delay* and *wait\_interval* delay parameters. We first observed network traffic oscillations and then we represented it by instability measurement of *average\_* $\delta_{instability}$  shown in equation 11. The simulation showed that the network traffic instability was decreased after such delays were added before the calculated optimal link weights were applied to the network. Consequently, we obtained link weights selection that yielded more stable measurement of network traffic with less oscillation.

The research results show that the hybrid approach can adapt to various network topologies such as Abilene and Polska, where both topology traffic matrices showed similar and consistent results for reducing MLU and PC. Adding *sequential\_delay* and

*wait\_interval* to the network system proved further the effectiveness of our hybrid approach, which yielded a new approach called *hybrid\_stable* approach. The *hybrid\_stable* approach combined both the hybrid approach together with using best *sequential\_delay* and *wait\_interval* delay parameters.

#### *Limitation of Network Topology Flow*

This study lacked the datasets for Polska topology that were required for testing our approaches (Gianoli, 2014). We created three synthetic different traffic flow datasets using MGEN that mimic low, medium, and high levels of traffic. We used similar simulation and methodology system such as Abilene network, that had already accessible datasets which are based on the study of (Tune et al., 2013). In the case of Abilene topology, we used existing data sets from 2004 described in Orlowski et al. (2010), therefore not having recent data sets for the use of academic research is considered a limitation for this network topology, and for other network topologies that are used in similar research.

#### **Implications**

The results of this research allow various recommendations to be considered by the network operator in order to save power consumption without increasing link utilization of the network. Changing the link weights drastically may result in various optimal solutions considered so that the network traffic is directed to different best solutions back and forth that can cause instability or network oscillation. The selection of the  $\Delta w$  value used in our simulation considers the tradeoff between performance and network instability.

This hybrid approach can fit other network topologies, and can be easily adapted to run in parallel with multiple CPUs, and thus reduce the runtime. The various approaches developed were all based on real time traffic datasets from 2004 for Abilene topology. Obtaining recent datasets for this topology or other similar topologies and using our best hybrid approach may yield better solutions.

In summary, we advanced current state of energy aware networks by the effort presented in this dissertation. The methods presented were able to reduce power consumption while not affecting MLU and thus enabling network operators to save network energy without affecting their network load balance.

### **Recommendations for Future Work**

The work conducted in this research extended the body of knowledge in the area of saving Internet energy while keeping the network load balance at accepted level (Addis et al., 2014; Amaldi et al., 2013; Francois et al., 2014; Zhang et al., 2010) . We have built this work upon several prior contributions made by other researchers (Francois et al., 2014; Zhang et al., 2010). It is likely that additional studies and research work will continue to improve on the work conducted in this investigation.

#### *Network Topology Flow Measurement and Online Traffic*

Future work would consider merging our approach with optimization of traffic pattern analysis and measurement. Our simulation provided a consistent and repeatable environment in order to compare different approaches using the same datasets of traffic patterns. The best way to show how a similar future approach can be effective in saving energy without affecting MLU is to run it autonomously on a real online traffic data.

Continuous analysis of online traffic is imminent and is required for the prevailing changing nature of Internet traffic.

#### *Larger Network Topologies with Real Time Traffic*

A second recommendation is to expand this approach for supporting larger network topologies. Larger network topologies exhibit a different challenge for converging the network flow after changing the link weights that may cause further oscillations and instability in the network. Therefore, further study into the algorithm's parameters, mutation, crossover, and sequential and wait interval delays is inevitable and may yield a superior approach for handling larger networks that comprise of many links and nodes.

#### *Adaptive GA Strategies of Multi Processing with Learning Methods*

A third concept that is extracted from using adaptive methods in our approach is to enhance the approach with multi processors that can utilize learning of best solutions. The algorithm will run a number of processes in parallel for a predefined time interval. In the end of each time interval, the best solution and the best GA selection method are shared among the various processes running with DEAP and then we apply the best solution that was learned to all processes. Thus, the GA selection strategy along with best solution will be used for the next time interval. The latter approach could achieve better optimal solutions for reducing PC and MLU due to the concurrent learning process.

#### *Further Instability Studies with Comprehensive and Variable Delay Parameters*

Lastly, is to enhance the instability study with respect to the delay parameters, and perform a more meticulous selection of those parameters and their effect on both

performance and instability of network traffic. By performance, we mean reducing MLU and PC. A number of research studies have improved the Internet routing and network monitoring unquestionably, such as discovery of slow convergence and persistent oscillation in routing protocols (da Silva & Mota, 2017; Roughan et al., 2011). Building upon our approach, a variable *sequential\_delay* can be applied for each nodes' links, which can be different from one node to another node depending on the amount of link weight change obtained after each iteration of the GA algorithm. The *wait\_interval* delay parameter is treated the same way as in our approach.

## Summary

Recent interest in research of energy aware networks have become evident and several authors proposed various approaches for reducing network energy while not affecting load balance (Addis et al., 2016; Amaldi et al., 2013; Capone et al., 2013; Francois et al., 2014; Zhang et al., 2010). This dissertation improved the reduction of network energy by using a multi-objective optimization technique while using real network traffic flow. The study used real network traffic for Abilene network topology from datasets published from 2004 for specific days; 5<sup>th</sup> of June, 5<sup>th</sup> of August, and 5<sup>th</sup> of September of the same year. For Polska topology, we used a synthetic traffic flow that was created by MGEN in a similar fashion such as Abilene topology but using three levels of traffic flow; low medium and high. For the case where many random multi-objective simulations were carried out, we performed averaging of the results (Francois et al., 2014). We collected both MLU and PC values based on our algorithm, using CORE network emulator and DEAP optimization Python package.

The best approach chosen was the hybrid approach that combined the random approach and the delta weight approach. Running the algorithm in a combined hybrid fashion returned better results and we were able to obtain between 30.95% to 42.86% reduction in MLU, and from 32.45% to 35.24% reduction in PC for Abilene network topology. Running the same hybrid approach on Polska topology yielded a reduction of MLU in the range of 18.52% to 35.14% and PC was decreased from 25% to 34.38%.

We introduced instability modeling and measurement method where we can measure the oscillation in the network by using equation 11, and we carried out various simulations in order to achieve least instability in traffic flow measurements for both network topologies. Our best approach called *hybrid\_stable* combined both the hybrid approach along with using best *sequential\_delay* and *wait\_interval* delay parameters giving further improvement in reducing MLU and PC, simultaneously using our DEAP algorithm. We showed a reduction of instability measurement from 175% down to 8.6% for Abilene network topology when using a value of 1sec for both *sequential\_delay* and *wait\_interval* delay parameters. For Polska network topology the instability measurement was reduced from 145.5% down to 27.17% when we used 1sec for *sequential\_delay* and *wait\_interval* delay parameters.

While this *hybrid\_stable* approach further reduced MLU and PC, proven by consistent results among the various network topologies there is still some room for improvement. It is still possible that future research can find better settings for the delay parameters that can achieve further stability in measuring the traffic flow.

## References

- Addis, B., Capone, A., Carello, G., Gianoli, L. G., & Sanso, B. (2012). *Energy-aware multiperiod traffic engineering with flow-based routing*. Paper presented at the 2012 IEEE International Conference on Communications (ICC).
- Addis, B., Capone, A., Carello, G., Gianoli, L. G., & Sanso, B. (2014). Energy management through optimized routing and device powering for greener communication networks. *IEEE/ACM Transactions on Networking (TON)*, 22(1), 313-325.
- Addis, B., Capone, A., Carello, G., Gianoli, L. G., & Sansò, B. (2016). Energy management in communication networks: a journey through modelling and optimization glasses. *Computer Communications*.
- Ahrenholz, J., Danilov, C., Henderson, T. R., & Kim, J. H. (2008). *CORE: A real-time network emulator*. Paper presented at the Military Communications Conference, 2008. MILCOM 2008. IEEE.
- Amaldi, E., Capone, A., & Gianoli, L. G. (2013). Energy-aware IP traffic engineering with shortest path routing. *Computer Networks*, 57(6), 1503-1517.
- Antonakopoulos, S., Fortune, S., & Zhang, L. (2010). *Power-aware routing with rate-adaptive network elements*. Paper presented at the GLOBECOM Workshops (GC Wkshps), 2010 IEEE.
- Bianzino, A. P., Chaudet, C., Larroca, F., Rossi, D., & Rougier, J.-L. (2010). *Energy-aware routing: a reality check*. Paper presented at the GLOBECOM Workshops (GC Wkshps), 2010 IEEE.
- Bianzino, A. P., Chaudet, C., Rossi, D., & Rougier, J. (2012). A survey of green networking research. *Communications Surveys & Tutorials, IEEE*, 14(1), 3-20.
- Bolla, R., Bruschi, R., Davoli, F., & Cucchietti, F. (2011). Energy efficiency in the future internet: a survey of existing approaches and trends in energy-aware fixed network infrastructures. *Communications Surveys & Tutorials, IEEE*, 13(2), 223-244.
- Capone, A., Cascone, C., Gianoli, L. G., & Sanso, B. (2013). *OSPF optimization via dynamic network management for green IP networks*. Paper presented at the Sustainable Internet and ICT for Sustainability (SustainIT), 2013.
- Celenlioglu, M. R., Goger, S. B., & Mantar, H. A. (2014). *An SDN-based energy-aware routing model for intra-domain networks*. Paper presented at the Software, Telecommunications and Computer Networks (SoftCOM), 2014 22nd International Conference on.

- Chabarek, J., Sommers, J., Barford, P., Estan, C., Tsiang, D., & Wright, S. (2008). *Power awareness in network design and routing*. Paper presented at the INFOCOM 2008. The 27th Conference on Computer Communications. IEEE.
- Chiaraviglio, L., Mellia, M., & Neri, F. (2012). Minimizing ISP network energy cost: formulation and solutions. *IEEE/ACM Transactions on Networking (TON)*, 20(2), 463-476.
- Christensen, K., Reviriego, P., Nordman, B., Bennett, M., Mostowfi, M., & Maestro, J. A. (2010). IEEE 802.3 az: the road to energy efficient ethernet. *IEEE Communications Magazine*, 48(11).
- Cianfrani, A., Eramo, V., Listanti, M., Marazza, M., & Vittorini, E. (2010). *An energy saving routing algorithm for a green OSPF protocol*. Paper presented at the INFOCOM IEEE Conference on Computer Communications Workshops, 2010.
- da Silva, R. B., & Mota, E. S. (2017). A survey on approaches to reduce BGP interdomain routing convergence delay on the Internet. *IEEE Communications Surveys & Tutorials*, 19(4), 2949-2984.
- Deb, K., Agrawal, S., Pratap, A., & Meyarivan, T. (2000). A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. *Lecture notes in computer science*, 1917, 849-858.
- Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *Evolutionary Computation, IEEE Transactions on*, 6(2), 182-197.
- Ericsson, M., Resende, M. G. C., & Pardalos, P. M. (2002). A genetic algorithm for the weight setting problem in OSPF routing. *Journal of combinatorial optimization*, 6(3), 299-333.
- Feamster, N., Rexford, J., & Zegura, E. (2013). The road to SDN. *Queue*, 11(12), 20.
- Fisher, W., Suchara, M., & Rexford, J. (2010). *Greening backbone networks: reducing energy consumption by shutting off cables in bundled links*. Paper presented at the Proceedings of the first ACM SIGCOMM workshop on Green networking.
- Fortz, B., & Thorup, M. (2000). *Internet traffic engineering by optimizing OSPF weights*. Paper presented at the INFOCOM 2000. Nineteenth annual joint conference of the IEEE computer and communications societies. Proceedings. IEEE.
- Francois, F., Wang, N., Moessner, K., Georgoulas, S., & Xu, K. (2014). On IGP link weight optimization for joint energy efficiency and load balancing improvement. *Computer Communications*, 50, 130-141.



- Garroppo, R. G., Giordano, S., Nencioni, G., & Scutellà, M. G. (2013a). Mixed integer non-linear programming models for green network design. *Computers & Operations Research*, 40(1), 273-281.
- Garroppo, R. G., Giordano, S., Nencioni, G., & Scutellà, M. G. (2013b). Power-Aware Routing and Network Design with Bundled Links: Solutions and Analysis. *Journal of Computer Networks and Communications*, 2013.
- Gelenbe, E., & Mahmoodi, T. (2011). *Energy-aware routing in the cognitive packet network*. Paper presented at the ENERGY 2011, The First International Conference on Smart Grids, Green Communications and IT Energy-aware Technologies.
- Gianoli, L. G. (2014). Energy-aware traffic engineering for wired IP networks. *Departement De G 'Enie' Electrique' Ecole Polytechnique De Montreal*.
- Gunaratne, C., Christensen, K., & Nordman, B. (2005). Managing energy consumption costs in desktop PCs and LAN switches with proxying, split TCP connections, and scaling of link speed. *International Journal of Network Management*, 15(5), 297-310.
- Gupta, M., & Singh, S. (2003). *Greening of the Internet*. Paper presented at the Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications.
- Heller, B., Seetharaman, S., Mahadevan, P., Yiakoumis, Y., Sharma, P., Banerjee, S., & McKeown, N. (2010). *ElasticTree: Saving Energy in Data Center Networks*. Paper presented at the NSDI.
- Idzikowski, F., Chiaraviglio, L., Cianfrani, A., Vizcaino, J. L., Polverini, M., & Ye, Y. (2016). A survey on energy-aware design and operation of core networks. *IEEE Communications Surveys & Tutorials*, 18(2), 1453-1499.
- Jain, H., & Deb, K. (2013). *An improved adaptive approach for elitist nondominated sorting genetic algorithm for many-objective optimization*. Paper presented at the International Conference on Evolutionary Multi-Criterion Optimization.
- Jakma, P., & Lamparter, D. (2014). Introduction to the quagga routing suite. *IEEE Network*, 28(2), 42-48.
- Mahadevan, P., Sharma, P., Banerjee, S., & Ranganathan, P. (2009). A power benchmarking framework for network devices. In *NETWORKING 2009* (pp. 795-808): Springer.
- Moulierac, J., & Phan, K. (2014). *Optimizing IGP link weights for energy-efficiency in a changing world*. INRIA,

- Nascimento, M. R., Rothenberg, C. E., Salvador, M. R., & Magalhães, M. F. (2010). *Quagflow: partnering quagga with openflow*. Paper presented at the ACM SIGCOMM Computer Communication Review.
- Nedevschi, S., Popa, L., Iannaccone, G., Ratnasamy, S., & Wetherall, D. (2008). *Reducing Network Energy Consumption via Sleeping and Rate-Adaptation*. Paper presented at the NSDI.
- Okonor, O., Wang, N., Sun, Z., & Georgoulas, S. (2014). *Link sleeping and wake-up optimization for energy aware ISP networks*. Paper presented at the 2014 IEEE Symposium on Computers and Communications (ISCC).
- Orlowski, S., Wessäly, R., Pióro, M., & Tomaszewski, A. (2010). SNDlib 1.0—survivable network design library. *Networks*, 55(3), 276-286.
- Rainville, D., Fortin, F.-A., Gardner, M.-A., Parizeau, M., & Gagné, C. (2012). *Deap: A python framework for evolutionary algorithms*. Paper presented at the Proceedings of the 14th annual conference companion on Genetic and evolutionary computation.
- Rexford, J. (2006). Route optimization in IP networks. In *Handbook of Optimization in Telecommunications* (pp. 679-700): Springer.
- Roughan, M., Willinger, W., Maennel, O., Perouli, D., & Bush, R. (2011). 10 lessons from 10 years of measuring and modeling the internet's autonomous systems. *IEEE Journal on Selected Areas in Communications*, 29(9), 1810-1821.
- Sakellari, G. (2009). The cognitive packet network: a survey. *The Computer Journal*, bxp053.
- Shaikh, A., Varma, A., Kalampoukas, L., & Dube, R. (2000). *Routing stability in congested networks: Experimentation and analysis*. Paper presented at the ACM SIGCOMM Computer Communication Review.
- Srinivas, N., & Deb, K. (1994). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary computation*, 2(3), 221-248.
- Tucker, R., Baliga, J., Ayre, R., Hinton, K., & Sorin, W. (2008). *Energy consumption in IP networks*. Paper presented at the ECOC Symposium on Green ICT.
- Tune, P., Roughan, M., Haddadi, H., & Bonaventure, O. (2013). Internet traffic matrices: A primer. *Recent Advances in Networking, 1*, 1-56.
- Vallet, J., & Brun, O. (2014). Online OSPF weights optimization in IP networks. *Computer Networks*, 60, 1-12.
- Varadhan, K., Govindan, R., & Estrin, D. (2000). Persistent route oscillations in inter-domain routing. *Computer Networks*, 32(1), 1-16.

- Vasić, N., Bhurat, P., Novaković, D., Canini, M., Shekhar, S., & Kostić, D. (2011). *Identifying and using energy-critical paths*. Paper presented at the Proceedings of the Seventh Conference on emerging Networking EXperiments and Technologies.
- Zeadally, S., Khan, S. U., & Chilamkurti, N. (2012). Energy-efficient networking: past, present, and future. *The Journal of Supercomputing*, 62(3), 1093-1118.
- Zhang, M., Yi, C., Liu, B., & Zhang, B. (2010). *GreenTE: Power-aware traffic engineering*. Paper presented at the Network Protocols (ICNP), 2010 18th IEEE International Conference on.